

The machine in the ghost: An Educational Design
Research study that explores the teaching of
Computational Thinking to Irish second-level
students

Colette Kirwan,

BSc., H. Dip. C.S., H. Dip. Software Eng., MSc.

School of STEM Education, Innovation and Global Studies

A thesis submitted in partial fulfilment of the requirements
of Dublin City University,
for the award of Doctor of Philosophy

Supervisors

Dr Enda Donlon, Dr Eamon Costello, and Prof. Mark
Brown

September 2021

Declaration

I hereby certify that this material, which I now submit for assessment on the programme of study leading to the award of Doctor of Philosophy is entirely my own work, and that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge breach any law of copyright, and has not been taken from the work of others save and to the extent that such work has been cited and acknowledged within the text of my work.

Signed:

ID No.: 16213976

Date: 08/9/2021

Publications associated with this research

Kirwan, C., Costello, E. and Donlon, E. (2018) ‘Computational Thinking and Online Learning: A Systematic Literature Review.’, in *Proceedings of the: 17th European Conference on eLearning, ECEL 2018*,. Athens, Greece: Academic Conferences and Publishing International Limited, pp. 650–657. Available at:
https://www.researchgate.net/publication/329443146_Computational_Thinking_and_Online_Learning_A_Systematic_Literature_Review

See Appendix A for a full list.

Acknowledgements

Working for a PhD is a personal journey. However, it is not a solo one. I would like to thank my PhD supervisors, Dr Eamon Costello and Dr Enda Donlon for their endless support, guidance and patience. Your encouragement kept me going.

I want to thank my husband, Tim, for his unwavering support, encouragement, and for being the best wingman possible. My children, Conal and Cora for their amazing understanding of my PhD obligations. I would like to thank my parents for instilling in me a love of learning.

I want to thank my participants: the teachers, students and content experts who made this study possible. I am still in awe of how you freely gave up your time and expertise to help me complete this journey.

I would also like to thank DCU's Institute of Education. I was there at your beginning and have benefitted from the support, facilities, and community you have put in place for PhD students. I would like to thank Prof Mark Brown and Dr Margaret Leahy for their encouragement and 'check in' chats.

I would like to thank my PhD colleagues Joan Whelan and Aoife Merrins for their support.

I want to thank my friend Laura Price, who can probably talk about Computational Thinking as well as I can now. Finally, I want to thank the Irish Research Council, whose funding made this incredible journey possible.

Table of Contents

Declaration	<i>i</i>
Publications associated with this research	<i>ii</i>
Acknowledgements	<i>iii</i>
Table of Contents	<i>iv</i>
List of Abbreviations	<i>viii</i>
List of Figures	<i>ix</i>
List of Tables	<i>xii</i>
Abstract	<i>xv</i>
1 Introduction	1
1.1 Chapter Introduction	1
1.2 The Educational Problem	2
1.3 The Solution	3
1.4 Unplugged Activities	4
1.5 Research Question	5
1.6 Thesis Structure	7
2 What is Computational Thinking?	10
2.1 Introduction	10
2.2 Computational Thinking: The Genesis	10
2.3 Computational Thinking: Problem Solving	12
2.4 Computational Thinking: Programming Languages	15
2.5 Computational Thinking in Irish Policy documents	16
2.5.1 Definitions	18
2.5.2 Applicability	18
2.6 Computational Thinking in the Irish media	20
2.7 Computational Thinking: Working towards a definition	22
2.8 Conclusion	24
3 Systematic Literature Review	25
3.1 Introduction	25
3.2 Methodology	25
3.2.1 Specification of the Research Question	26
3.2.2 Development of a Review Protocol	27
3.2.3 Identification of Research	27
3.2.4 Selection of primary studies	28
3.2.5 Inclusion Criteria.....	29
3.2.6 Exclusion Criteria	29
3.2.7 Study Quality Assessment	31
3.2.8 Data extraction and monitoring	32
3.3 Literature Review	33
3.3.1 What are the current pedagogical approaches to teaching Computational Thinking online?.....	34
3.3.2 Literature Review Findings	46
3.4 Conclusion	49
4 Methodology: Educational Design Research	50
4.1 Introduction	50
4.2 Worldview: Pragmatism	50
4.2.1 Pragmatism Rationale	51
4.2.2 Dewey: Theory of Knowledge and Inquiry	51
4.3 Research Approach	53

4.3.1	Educational Design Research.....	54
4.3.2	Motivation and Justification for EDR.....	55
4.3.3	Design Research Limitations.....	57
4.3.4	Mapping: Conjecture Map.....	59
4.4	Design Research Process	62
4.4.1	Design Research: Type.....	64
4.4.2	Design Research: Phases	65
4.4.3	Preliminary Analysis	67
4.4.4	Prototype phase	68
4.4.5	Semi Summative Phase	75
4.5	Data Collection: Evaluations.....	76
4.5.1	Screening.....	80
4.5.2	Questionnaires	82
4.5.3	Pre and Post Tests	87
4.5.4	Journals/Emails.....	89
4.5.5	Researcher Notes	90
4.5.6	Teachers Interviews.....	90
4.5.7	Survey	92
4.5.8	Student Artefacts.....	93
4.6	Data Collection: Quality	94
4.6.1	Content Expert Interviews.....	95
4.7	Data Analysis	96
4.7.1	Stage 1 'On the fly' Analysis	96
4.7.2	Stage 2 Validation Analysis.....	97
4.7.3	Stage 3 Thematic Analysis	99
4.7.4	Questionnaires	101
4.7.5	Pre and Post Tests	107
4.8	Ethical Consideration	109
4.8.1	Avoid harm to participants.....	110
4.8.2	Ensure informed consent	110
4.8.3	Privacy	110
4.8.4	Deception	111
4.9	Summary.....	111
5	<i>Computational Thinking Course and Instructional Design</i>	112
5.1	Quality	112
5.1.1	Validity.....	112
5.1.2	Practicality	113
5.1.3	Effectiveness.....	113
5.2	Engagement	113
5.2.1	How is engagement defined?	114
5.3	Low threshold.....	115
5.3.1	Unplugged Activities.....	115
5.4	How will the Computational Course be designed?	119
5.4.1	The Synthesis of ADAPTTER.....	119
5.5	Irish Educational System	124
5.6	What topics should the Computational Thinking course cover?.....	124
5.6.1	Course Materials	125
5.7	Computational Thinking Course Version 1	125
5.7.1	Unit 1: Problem Solving and Computational Thinking.....	126
5.7.2	Unit 2: Algorithms and Logic:	132
5.7.3	Unit 3: Common Algorithms:.....	137

5.7.4	Unit 4: Ethics and Artificial Intelligence (AI)	140
5.7.5	Unit 5: Fundamentals of Programming	147
5.8	Summary	150
6	<i>Prototype Phase Version 1.....</i>	<i>152</i>
6.1	Introduction.....	152
6.2	Research Design Evolution	152
6.3	Recruitment.....	154
6.3.1	Recruitment Policy Implementation	156
6.3.2	Teacher Recruitment	156
6.4	Data Analysis	160
6.5	Revisions	161
6.6	Evaluations	205
6.6.1	Effectiveness.....	205
6.6.2	Engagement (Teachers' Perspective)	248
6.6.3	Content.....	254
6.6.4	Pedagogy	255
6.7	Chapter Summary	257
7	<i>Prototype Phase Version 2.....</i>	<i>259</i>
7.1	The Setting	260
7.1.1	School Details	260
7.2	Workshop.....	260
7.2.1	Workshop Findings.....	261
7.2.2	Effectiveness.....	262
7.2.3	Low threshold.....	263
7.2.4	Course Content.....	263
7.2.5	Changes to content	264
7.3	Finding from Version 2: Teachers' perspectives.....	264
7.3.1	Theme 1: Teachers' Experience.....	265
7.3.2	Theme 2: Learning Environment:	276
7.3.3	Theme 3: Students' Experiences	278
7.3.4	Theme 4: Course Design.....	281
7.4	Chapter Summary	297
8	<i>Semi-Summative Phase Version 3.....</i>	<i>299</i>
8.1	Introduction.....	299
8.2	Semi-Summative Phase	299
8.3	Student Profile.....	300
8.4	Content Experts	300
8.5	Findings	300
8.5.1	Engagement.....	301
8.5.2	Low threshold.....	305
8.5.3	Effectiveness.....	307
8.5.3.1.3	Recommendations	312
8.5.4	Quality	325
8.5.5	Summary	327
9	<i>Conclusion</i>	<i>328</i>
9.1	Implications for Irish Educational Policy.....	331
9.2	Implications for Irish Curriculum.....	333
9.3	Implications for Irish Initial Teacher Education.....	333
9.4	ADAPTER Framework and Guidelines.....	335

9.5	Implications for Practice.....	341
9.6	Limitations	341
9.7	Future Work	342
9.8	Conclusion.....	343
References		344
Appendices		366
Appendix A: List of Conference Proceedings.....		367
Appendix B: Engagement Questionnaire		368
Appendix C: End of Course Questionnaire.....		369
Appendix D: Pre and Post Tests (Version 1)		372
Appendix E: MCQ Tests.....		375
Appendix F: Course Glossary		382
Appendix G: Summary 'On The Fly' Revisions.....		384
Appendix H: Engagement scores Version 1		386
Appendix I: Revisions from Validation Stage.....		389
Appendix J: Pre and Post Test Rubrics (Version 1).....		391
Appendix K: Logic and Ethics Results (Version 1)		394
Appendix L: Original Assessment Plans.....		396
Appendix M: Project Assessment.....		406
Appendix N: Workshop Questionnaire and Data.....		407

List of Abbreviations

ADAPTTER	Activities, Demonstration, Application, Pre-activation, Theory, Transparency, Exemplification, Reflection
CAS	Computing At School
CESI	Computers in Education Society of Ireland'
CSTA	Computer Science Teachers Association
CT	Computational Thinking
DCU	Dublin City University
DCYA	Department of Children and Youth Affairs
DES	Department of Education and Skills
ECT	Exploring Computational Thinking
EDR	Educational Design Research
IFIP	International Federation for Information Processing
ISTE	International Society for Technology in Education
MOOC	Massive Open Online Course
NCCA	National Council for Curriculum and Assessment

List of Figures

Figure 1-1 Thesis structure and outline	7
Figure 2-1 Barefoot Computing diagram of Computational Thinking (Barefoot 2020) Reproduced with permission	14
Figure 2-2 The Nexis Database	20
Figure 2-3 Contextual and Morphemic cues used to surmise Computational Thinking definition.....	21
Figure 2-4 Policy Documents mentioned in Irish Media.....	22
Figure 2-5 ‘Agreement and disagreement around two views of what Computational Thinking should be.’ (Curzon et al., 2019, p. 515) Copyright The Cambridge University Press. Reproduced with permission of The Cambridge University Press through PLSclear.	23
Figure 3-1 Qiqqa Reference Tool	30
Figure 3-2 Study selection process (Moher et al., 2009). Original printed (Kirwan, Costello and Donlon, 2018)	31
Figure 3-3 A screenshot of a snippet of the Google extraction form used to record information about the reviewed articles.....	33
Figure 3-4 Zones of Proximal Flow diagram where ZPD is located between regions of flow and anxiety ‘Republished with permission of ACM, from Basawapatna et al., 2013; permission conveyed through Copyright Clearance Center, Inc’	37
Figure 3-5 A screen capture of the MakeWorld website at makeworld.eu	40
Figure 4-1 Dewey’s model of inquiry Morgan (2014) (Reprinted with permission)	53
Figure 4-2 Reeves (2006) Design research approaches in educational technology research (Reproduced with permission of The Licensor through PLSclear.)	55
Figure 4-3 Conjecture Map for the Study	61
Figure 4-4 Hybrid Map that illustrates the design trajectory of the study and pertinent EDR activities	63
Figure 4-5 Phases and Activities that occurred during the EDR study (diagram format influenced by Mafumiko (2006) and Masole (2011))	66
Figure 4-6 Illustration of the three phases of EDR	67
Figure 4-7 Prototype Phase of the EDR study	68
Figure 4-8 Development of Prototype phase (Nieveen and Folmer, 2013) (copyright CC by attribution)	69
Figure 4-9 Overview of Version One	70
Figure 4-10 Gantt Chart showing the timeline of the lessons.....	72
Figure 4-11 Diagram highlighting the microcycles in the development approach (Gravemeijer and Cobb, 2006) (copyright CC by attribution)	73
Figure 4-12 EDR process diagram. The involvement of the Content Experts in Version 1	73
Figure 4-13 Prototype Phase Version 2	74
Figure 4-14 Version 3 of the intervention	75
Figure 4-15 Evaluation matchbook developed by Nieveen, Folmer and Vligen (2012)	78
Figure 4-16 Snapshot of the post-lesson engagement questionnaire	83
Figure 4-17 Snippet of ‘End of Course’ Questionnaire showing demonstration items....	84
Figure 4-18 Snippet of the End of Course Questionnaire showing application items....	85
Figure 4-19 Snippet of the End of Course Questionnaire showing items related to unplugged activities	85
Figure 4-20 Snapshot of the MCQ test given to students online.....	89
Figure 4-21 A screenshot of the questionnaire	93
Figure 4-22 Student Artefact from Unit 2: Writing an algorithm to draw a smiley face emoji	94

Figure 4-23 A snippet of the table devised from Expert 1 and Expert 2. This table contains the data categorised under the headings of Quality with the created subcode.....	99
Figure 4-24 A screenshot of the frequency statistics captured for school 1 in week 3.	102
Figure 4-25 A screenshot showing a scanned questionnaire and the created categories in NVivo.....	103
Figure 4-26 A screenshot illustrating the in vivo coding scheme adopted for the open-ended questions	105
Figure 4-27 The subcategories for the main category of Strengths.	105
Figure 4-28 The Setup sub-category.....	106
Figure 4-29 Categorised student answers to the question ‘What do you think Computational Thinking might mean?’	108
Figure 4-30 Rubric for grading the question ‘What do you think Computational Thinking might mean?’	109
Figure 5-1 Merrill’s First Principles of Instructional Design (Merrill, 2002)	120
Figure 5-2 Cover of the Teacher Guide	126
Figure 5-3 Slide taken from the PDST Leaving Certificate National Workshop 3, Session 3 (2019).....	127
Figure 5-4 Slide used to highlight pattern matching in Computer Science	131
Figure 5-5 The emoji that students re-create using instructions.	136
Figure 5-6 Screenshot of the http://20q.net/ site.....	143
Figure 5-7 Screenshot of Moral Machine (Scalable Cooperation and MIT Media Lab, no date).....	146
Figure 5-8 A slide showing a picture of the front cover of Hannah Fry’s book with the questions on the back cover (Fry, 2018).	147
Figure 5-9 A snippet of cheat sheet used in Week 5. Cheatsheet adapted from OCR GCSE in Computer Science Specification (Oxford Cambridge RSA (OCR), 2020)	150
Figure 6-1 Research Approach	153
Figure 6-2 Original Design Approach	153
Figure 6-3 Re-design of Research Approach for my Design Research Study	158
Figure 6-4 The emerging instructional framework	175
Figure 6-5 School 3 Median values	185
Figure 6-6 School 3 Mode Engagement values	185
Figure 6-7 A collage showing the creation of scripted chatbots.....	187
Figure 6-8 Initial themes created for the content experts analysis (NVivo 1.4.1)	191
Figure 6-9 Language theme expanded (NVivo 12)	191
Figure 6-10 Mindmap of the Content Experts Themes.....	192
Figure 6-11 Computational Flow Chart replaced with CT components in a circle	195
Figure 6-12 Essential Characteristics of algorithms Slide, before and after change	196
Figure 6-13 Median Values of the Satisfaction and BEST quality items	208
Figure 6-14 Mode Values of the Satisfaction and BEST quality items	208
Figure 6-15 Student (S14) partial answer to: ‘What are the strengths of this course?’	213
Figure 6-16 Student (A13): ‘What are the strengths of this course?’	213
Figure 6-17 Student (N20) answer to: ‘What are the strengths of this course?’	214
Figure 6-18 Median Values of items related to Unplugged Activities	215
Figure 6-19 Mode Values of items related to Unplugged Activities	216
Figure 6-20 Student (N7) answer to: ‘What are the strengths of this course?’	217
Figure 6-21 In Vivo answers coded under the Pedagogy category for School 2.....	218
Figure 6-22 Mode and Median values for demonstration items per school.....	219
Figure 6-23 The four themes generated from the analysis of the teacher data	221
Figure 6-24 Screenshot NVivo showing codes for Teacher Reactions	222
Figure 6-25 Mode and Median values for learning items per school.	226
Figure 6-26 Student S5 answer to ‘What do you think you learned during the course?’	232

Figure 6-27 Student S3 answer to ‘What do you think you learned during the course?’	233
Figure 6-28 Student S18 answer to ‘What do you think you learned during the course?’	234
Figure 6-29 Student N13 answer to ‘What do you think you learned during the course?’	235
Figure 6-30 Student A4 answer to ‘What do you think you learned during the course?’	235
Figure 6-31 NVivo Screenshot of themes related to the teachers’ perspective (V2) ...	249
Figure 6-32 NVivo capture of the Validation codes for Engagement	250
Figure 6-33 NVivo screenshot of sub-theme Pedagogy.....	253
Figure 6-34 NVivo Screenshot of Content Theme	254
Figure 6-35 NVivo nodes illustrating the codes captured under the Characteristics category	256
Figure 7-1 An NVivo Hierarchy chart generated from the workshop open-ended questions	261
Figure 7-2 Snippet from a teacher’s answer.	264
Figure 7-3 An Nvivo Screenshot displaying the four themes generated from the teacher qualitative data.....	265
Figure 7-4 NVivo screenshot of teachers’ quotes in relation to the course.....	273
Figure 7-5 A collage of Students’ answers for a Computational Thinking project issued at School 2 by Teacher 2	281
Figure 8-1 Median Engagement scores for School 1 (V3).....	301
Figure 8-2 Mode Engagement scores for School 1 (V3).....	302
Figure 8-3 Median Engagement scores for School 6.....	303
Figure 8-4 Mode Engagement scores for School 6.....	303
Figure 8-5 NVivo screenshot of codes captured from Teacher 11’s interview	304
Figure 8-6 Median values from School 1 (V3) and School 6 showing Satisfaction and BEST Quality scores.....	308
Figure 8-7 Mode values from School 1 (V3) and School 6 showing Satisfaction and BEST Quality scores.	308
Figure 8-8 Bar Chart of students’ responses to ‘Media used in the course was helpful to learning’	312
Figure 8-9 NVivo Screenshot showing the codes from the Teacher Reaction Theme for Version 3.....	314
Figure 8-10 Median Values from School 1 (V3) and School 6 showing items related to learning	316
Figure 8-11 Mode Values from School 1 (V3) and School 6 showing items related to learning	316
Figure 8-12 Pie chart showing results from School 6 to the statement: I see how I can apply what I learned in this course to other subjects (N=15)	317
Figure 8-13 Box Plot of Mean scores pre and post course for School 6	321
Figure 8-14 Paired Sample t Test for School 6.....	321
Figure 9-1 Thesis structure and outline	328
Figure 9-2 The ADAPTTER framework for developing a high quality, effective, low threshold, engaging, and practical course for teaching Computational Thinking	336

List of Tables

Table 2-1 Irish Policy Database Information and no. of documents returned from each database.....	17
Table 3-1 Search string composed of synonyms used in the systematic literature review ..	28
Table 3-2 Academic Database Information and no. of articles returned from each database	28
Table 3-3 List of articles that use visual programming languages to teach Computational Thinking	34
Table 3-4 List of articles that document playing a video game to teach Computational Thinking	39
Table 3-5 List of articles that use non-visual programming languages to teach Computational Thinking	42
Table 3-6 List of articles that use un-plugged activities to teach Computational Thinking	43
Table 3-7 List of articles that use competitions to teach Computational Thinking	44
Table 3-8 List of Collaboration Tools.....	45
Table 3-9 List of articles that form their own category	45
Table 4-1 Outputs from this study	65
Table 4-2 Version 1 timeline	70
Table 4-3 Version 2 timeline and details	75
Table 4-4 Timeline and details of the implementation of Version 3	76
Table 4-5 A summary of the evaluation activities and corresponding instruments.....	79
Table 4-6 Checklist developed for Merrill’s First Principles of Instructions	80
Table 4-7 Snippet of the Learning Outcome description table for Lesson 1	81
Table 4-8 Snippet of the assessments developed for Lesson 1	81
Table 4-9 Snippet of the table that maps assessments and learning outcomes with Bloom’s Taxonomy	81
Table 4-10 Question 1 and Question 3 issued in the pre and post test for Version 1 of the course	88
Table 4-11 Instruments used to gather data on course quality.....	94
Table 4-12 Categories used in the Validation Stage Analysis	97
Table 4-13 List of Actions taken to improve the Content of Version 1 of the course	99
Table 4-14 Descriptions, Examples and Frequencies of the Strengths sub-category for a participating school.	106
Table 4-15 Results and grades from the analysis of the open-ended question ‘What do you think Computational Thinking might mean?’	108
Table 5-1 Theoretical support for ADAPTTER	121
Table 6-1 Teacher Profile from participating schools	158
Table 6-2 Summary of school types and dates for participating schools.....	159
Table 6-3 Content and Instructional changes to Unit 1	163
Table 6-4 Content and Instructional changes to Unit 2	167
Table 6-5 Content and Instructional changes to Unit 3	170
Table 6-6 Content and Instructional changes to Unit 4	176
Table 6-7 Content and Instructional changes to Unit 5	180
Table 6-8 Cross Code Analysis table for Expert 1 and 2.....	190
Table 6-9 Results from the analysis of the open-ended question: Do you have any specific recommendations for improving the course? (N=14, School 1).....	201
Table 6-10 Results from the analysis of the open-ended question: Do you have any specific recommendations for improving the course? (N=22, School 2).....	202
Table 6-11 Results from the analysis of the open-ended question: Do you have any specific recommendations for improving the course? (N=26, School 3).....	203

Table 6-12 Results from the analysis of the open-ended question: Do you have any specific recommendations for improving the course? (N=8, School 4).....	204
Table 6-13 Instruments used to gather data on effectiveness	206
Table 6-14 Summary Table of the number of students participating from each school....	207
Table 6-15 Results from the analysis of the open-ended question: What are the strengths of this course? (N=14, School 1).....	209
Table 6-16 Results from the analysis of the open-ended question: What are the strengths of this course? (N=22, School 2).....	210
Table 6-17 Results from the analysis of the open-ended question: What are the strengths of this course? (N=26, School 3).....	211
Table 6-18 Results from the analysis of the open-ended question: What are the strengths of this course? (N=9, School 4).....	211
Table 6-19 Results from the analysis of the open-ended question: ‘What do you think you learned during the course?’ (N=14, School 1).....	227
Table 6-20 Results from the analysis of the open-ended question: What do you think you learned during the course? (N=22, School 2)	227
Table 6-21 Results from the analysis of the open-ended question: What do you think you learned during the course? (N=26, School 3)	229
Table 6-22 Results from the analysis of the open-ended question: What do you think you learned during the course? (N=9, School 4)	230
Table 6-23 Bar charts depicting the category of answers and their frequency for CT Q pre test.....	238
Table 6-24 Bar charts depicting the category of answers and their frequency for CT Q post test.....	240
Table 6-25 Results from the analysis of the open-ended question (Pre-Test) What do you think Computational Thinking might mean? (N=15, School 1)	241
Table 6-26 Results from the analysis of the open-ended question (Post-Test) What do you think Computational Thinking might mean? (N=13, School 1)	241
Table 6-27 Bar charts depicting the category of answers and their frequency for Algorithm Question pre test.	243
Table 6-28 Bar charts depicting the category of answers and their frequency for Algorithm Question post test.....	244
Table 6-29 Results from the analysis of the open-ended question (Pre-Test) Explain what you understand by algorithms? (N=30, School 3)	245
Table 6-30 Results from the analysis of the open-ended question (Post-Test) Explain what you understand by algorithms? (N=26, School 3)	245
Table 7-1 Summarises the feedback from Unit 1	283
Table 7-2 Summarises the feedback from Unit 2	287
Table 7-3 Summarises the feedback from Unit 3	289
Table 7-4 Summarises the feedback from Unit 4	291
Table 7-5 Summarises the feedback from Unit 5	294
Table 8-1 School profiles for Version 3	300
Table 8-2 Results from the analysis of the open-ended question: What are the strengths of this course? (N=15, School 6).....	309
Table 8-3 Results from the analysis of the open-ended question: What are the strengths of this course? (N=5, School 1).....	310
Table 8-4 Results from the analysis of the open-ended question: Do you have any specific recommendations for improving the course? (N=15, School 6).....	313
Table 8-5 Results from the analysis of the open-ended question: What do you think you learned during the course? (N=15, School 6)	318
Table 8-6 Results from the analysis of the open-ended question: What do you think you learned during the course? (N=5, School 1(V3)).....	320

Table 8-7 A snapshot of the pre and post MCQ test scores for School 6323

Abstract

The machine in the ghost: An educational design research study that explores the teaching of Computational Thinking to Irish second-level students

Colette Kirwan BSc, H Dip CS, H Dip Software Eng, MSc

Computational Thinking is a problem-solving process that draws on concepts fundamental to Computer Science. These concepts can support problem-solving across many disciplines. The Digital Strategy for Schools (2015-2020) describes the Irish Government's intention to give every student in compulsory education the opportunity to learn Computational Thinking.

This research is an Educational Design Research study underpinned by a pragmatic approach and concerned with Computational Thinking. It aims to answer the following question: what are the characteristics of a practical, engaging, effective, high quality, and low threshold course for both the learning and teaching of Computational Thinking to Irish post-primary teachers and students? This study also aims to validate whether unplugged activities can be successfully used to teach Computational Thinking.

This research study had three phases: preliminary analysis, prototype, and semi-summative. It was conducted in six schools with eleven teachers, four content experts, and over four hundred and forty six students. Data was gathered using various means: interviews, focus groups, teacher diaries, students' questionnaires, and students' artefacts. The analytic approach was mixed; it involved content and thematic analysis as well as descriptive statistics.

This study found that the following characteristics: activities, demonstration, application, pre-activation, transparency, theory, exemplification, and reflection (ADAPTTER) gave rise to a practical, engaging, effective, high quality, and low threshold Computational Thinking course. This study validated the use of unplugged activities as a pedagogy for teaching Computational Thinking.

1 Introduction

I had a bit of a light bulb moment after the course yesterday. Before yesterday when I was teaching computational thinking last year I was coming at it solely from a narrow computer programmer point of view and didn't fully grasp what it was. However, the layout of the course yesterday whereby computational thinking was outlined in non computer terms first made me realise something. In my previous life I worked on a large IT systems project ... My role was to map out all the current processes to be put into a specification for the IT programmers to programme. I now realise what I was doing for those three years was computational thinking - involving identifying patterns, decomposition and abstraction and culminating in spending nine months working solely on an allocation algorithm. The language of computational thinking wasn't used (other than algorithms) and I had never made the connection before. So I've gone from a very fluffy sense of what computational thinking is to really knowing what it's about. The key for me is that this is a skill that is much broader than a 'computer programmer' skill and this insight will completely change how I approach teaching this in the future and the confidence I have in teaching it. (Teacher 7, 2019, email after workshop)

1.1 Chapter Introduction

This first chapter of my thesis introduces my research, an Educational Design Research (EDR) study, whose primary function was to develop, design, and gather data on the efficacy of a Computational Thinking (CT) course for Irish post-primary students and teachers.

The mention of the methodology is important. EDR defined the research approach, informed the research questions, impacted this study's outcome and influenced the structure of this thesis.

Educational design studies start with defining an educational problem followed by a proposed solution (McKenney and Reeves, 2012). This introductory chapter follows this same path. It starts by clarifying the nature of an educational problem, teaching Computational Thinking to Irish teachers and second-level students. This is followed by a proposed solution, a Computational Thinking course and a set of design guidelines for its development. A core teaching approach of this PhD study, unplugged activities, is next introduced. This is followed by the research aims and an outline of the structure of this thesis.

1.2 The Educational Problem

The educational problem concerns Computational Thinking and how best to teach it to Irish teachers and Irish second-level students. Computational Thinking is a problem-solving process. It concerns concepts fundamental to Computer Science, for example, decomposition, pattern recognition, abstraction and algorithms. These concepts can be used to support problem-solving across all disciplines (not just Computer Science), for example, science, mathematics, and the humanities (Mohaghegh and McCauley, 2016).

Coupled with these problem-solving skills is the knowledge of how computing algorithms can be used and abused. Our world is both physical and digital. Students would benefit from understanding how this digital world works, specifically how algorithms drive it (Curzon *et al.*, 2019). They would also benefit from knowing how algorithms are used to manipulate how information is presented online in their daily lives. Computational Thinking can provide students with these skill sets (Buitrago Flórez *et al.*, 2017).

For these reasons (and more), Computational Thinking is considered by many to be a highly valuable 21st-century skill (Wing, 2006). It has gained the attention of many European government policymakers, including the Irish government (Bocconi *et al.*, 2016). The Irish Digital Strategy for Schools (2015-2020) outlines the Irish Government's intention to give every Irish student at both primary and post-primary the opportunity to learn Computational Thinking (DES, 2015).

Currently, in Ireland, in second level education, Computational Thinking is formally embedded into two subjects, the Junior Cycle short course Coding and the Leaving Certificate Computer Science subject (National Council for Curriculum and Assessment (NCCA), 2016; NCCA and DES, 2018). However, the rollout of these subjects is not without problems. The Oireachtas Library and Research Service Report (2017) on the

‘Introduction of Coding and Computer Science to the Irish curriculum’ highlight the following:

concerns have been expressed that schools may not have the time, resources, or expertise to roll out coding and computer science at second level;

there is little pedagogical research to guide teachers and those designing curriculum (2017, p. 1)

Information from the Department of Education and Skills (DES) presents the following picture. There are 715 post-primary schools in Ireland (DES, 2019). Regarding state examination subjects, coding has been a part of the Junior Cycle since 2014. Ten schools undertook coding as a Junior Cycle short course in the 2016-2017 year (DES, 2018a) and forty schools in the 2017-2018 year (DES, 2019). Computer Science was piloted as a Leaving Certificate subject in forty schools in 2018-2020. It is available to all schools from 2020-2021, if there are available teachers. Computational Thinking is currently not a topic that many Irish post-primary teachers formally teach.

1.3 The Solution

To aid in the effort of teaching Computational Thinking to post-primary students, this research designed, developed and evaluated a Computational Thinking course. This course introduced students and teachers to the problem-solving framework of Computer Science (which can be applied to other subjects). It facilitated the development of students’ logical and algorithmic thinking skills. It enabled students to appreciate the impact computers have on their lives through artificial intelligence and ethics. It also introduced some tenets of Computer Science, such as search algorithms, coding fundamentals and how computer algorithms work. The course content aligned with the Leaving Certificate Computer Science curriculum (where feasible) but had a low threshold entry point (minimal resources and minimal prerequisite knowledge). Students followed a ‘learn by doing’ framework. They learned Computational Thinking by engaging in activities that did not

initially involve a computer (unplugged), such as games, puzzles and thought experiments. Finally, students and teachers learned a language that they could use to articulate their thinking process.

1.4 Unplugged Activities

A core pedagogical strategy used to teach Computational Thinking in this study was unplugged activities. The term is associated with 'Computer Science Unplugged', which refers to activities that expose students to concepts and ideas from Computer Science without the use of a computer. Games, magic tricks, storytelling are examples of such activities. The activities are characterised as engaging, simple to implement and explain, and incorporate collaboration (Bell *et al.*, 2009). Lesson plans associated with these activities are available free online from the following sites: csunplugged.org, csunplugged.mines.edu and teachinglondoncomputing.org. Computer Science Unplugged activities also have the advantage of bridging the gap for teachers who are not proficient in Computer Science but are expected to teach it (Rodriguez, Rader and Camp, 2016). The rationale for deciding on the use of unplugged activities is manifold, and is discussed in detail in Chapter 4, but suffice to say, it is an approach favoured by teachers (Sentance and Csizmadia, 2017). Furthermore, it allows complex problems and topics to be addressed without students needing to learn to program first (Bell *et al.*, 2009).

There are also gaps in knowledge highlighted with respect to unplugged activities (Waite, 2017). Areas of research suggested are:

- how teachers are successfully embedding unplugged activities to teach computational thinking;

- how teachers are successfully embedding unplugged activities to teach computer science concepts;

- how best to use unplugged activities, and the effectiveness of the different types of unplugged activities, including how they should be best combined with other approaches such as in teaching programming. (Waite, 2017, p. 48)

This research investigates 1) how to successfully embed unplugged activities to teach Computational Thinking 2) how best to use unplugged activities, and 3) the effectiveness of the different types of unplugged activities.

1.5 Research Question

This PhD study began with lofty goals. Unplugged activities from Computer Science (Bell, Witten and Fellows, 1998) would be used to develop and teach an online Computational Thinking course, aimed at Irish teachers. The course would then be evaluated to ascertain how successful it was in empowering said teachers to teach Computational Thinking to Irish second-level students.

During the early stages of this research, it became clear that these original goals were not feasible in the proposed timeframe. This was for many reasons: the available research literature provided limited guidance, teachers' knowledge of Computational Thinking was poor, and teaching and pedagogical expertise were unclear to both teachers and researchers (see Chapter 2, 3). It was also apparent that this course could not be made available online to teach teachers until the efficacy of its pedagogy and materials was first proven in the classroom. For this to happen, answers to the following questions needed to be explored and discovered.

- What is the current baseline of knowledge for Irish Transition year students regarding Computational Thinking and Computer Science?
- Would unplugged activities work as a teaching approach with Irish second-level teachers in the classroom? Research has shown that unplugged activities have their origins as part of outreach programmes and are geared at primary school students (Bell, Witten and Fellows, 1998; Rodriguez *et al.*, 2017). Surprisingly, few empirical studies have been conducted about how Computer Science unplugged activities have been used in a regular classroom (Bell and Vahrenhold, 2018).
- What topics should the course include?

- Would teachers need pre-requisite knowledge to teach it?
- Would students find the course engaging and interesting?
- How should such a course be designed?
- What characteristics should the course have?

The above questions helped to formulate the research question that this course set out to answer, and also helped in deciding on the most appropriate methodology to use.

Kelly (2006) articulates the following:

in areas where little is known (for example, how to teach and how students learn statistics), exploratory or descriptive work naturally precedes (and informs) randomized field trials, which incidentally, are meaningless without this foundational work. (2006, p. 114)

He further outlines that design research methodology is recommended when teachers' knowledge is unsatisfactory, content knowledge is new, and teaching and pedagogical expertise are unclear (to both teachers and researchers) (Kelly, 2013). Design research is defined as research whose purpose is:

to design and develop an intervention (such as programs, teaching-learning strategies and materials, products and systems) as a solution to a complex educational problem as well as to advance our knowledge about the characteristics of these interventions and the processes to design and develop them, or alternatively to design and develop educational interventions (about for example, learning processes, learning environments and the like) with the purpose to develop or validate theories (Plomp, 2013, p. 15)

Plomp stresses that design research is not solely concerned with designing and developing an intervention. The researcher must also endeavour to discover design principles (or an intervention theory) applicable in a specific context. Therefore, research questions should adhere to the following heuristic:

‘What are the characteristics of an <intervention X> for the purpose/outcome Y in context Z?’ (Plomp, 2013, p. 27)

This research aims to answer the following question: **what are the characteristics of a high quality, practical, engaging, effective, and low threshold course for both the learning and teaching of Computational Thinking to Irish second-level teachers and students?**

This PhD study also aims to validate whether unplugged activities can be successfully used to teach Computational Thinking.

1.6 Thesis Structure

The structure of this thesis follows the three phases of EDR, which occurred in chronological order: preliminary analysis, prototype and semi-summative.

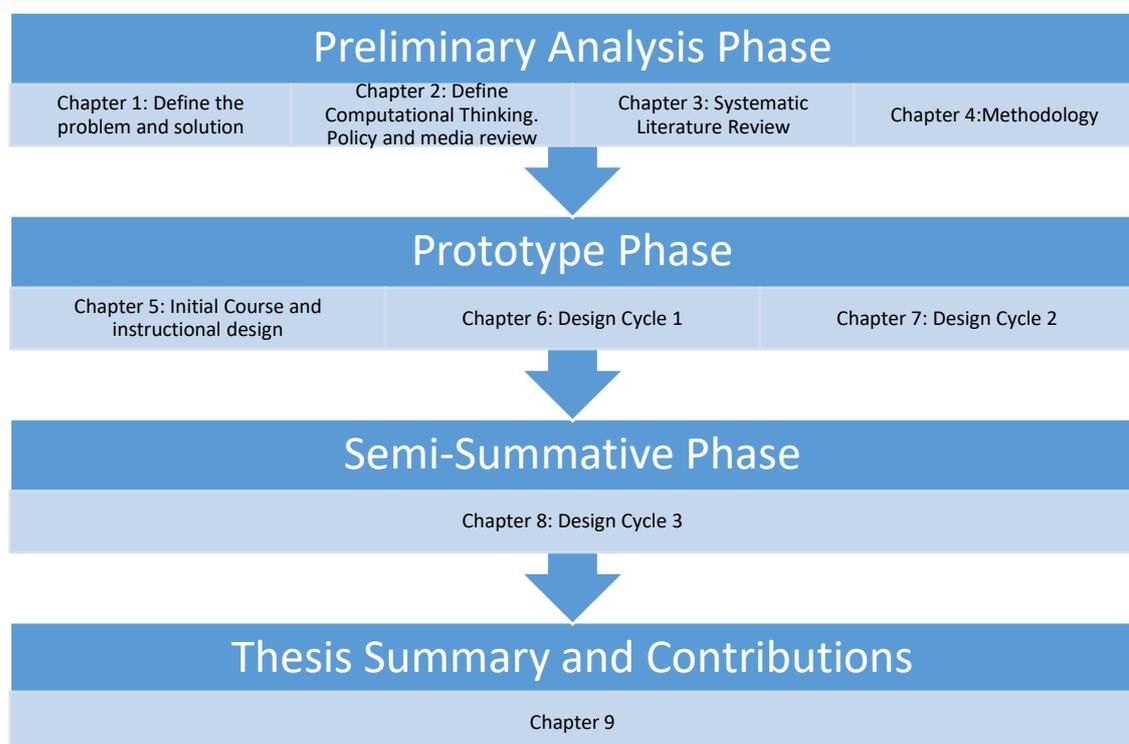


Figure 1-1 Thesis structure and outline

The Preliminary Analysis phase was concerned with gaining an insight into ‘the educational problem’. This involved understanding the current situation, which encompassed how Computational Thinking is defined in academic, Irish media, and policy documents. It involved a literature review on pedagogical strategies related to

Computational Thinking, and finally, research into the use of unplugged activities. These activities served to inform potential approaches to the proposed design and development of the solution, a Computational Thinking course.

Chapter 1 introduces the research. It outlines the educational problem that this study addresses, accompanied by its proposed solution. A brief review of core components of this research is provided, such as Computational Thinking, unplugged activities and EDR.

Chapter 2 serves to document the genesis, scope and applicability of Computational Thinking. An overview of how Computational Thinking is defined in academic literature, Irish policy and curriculum documents, and the Irish media is provided.

Chapter 3 provides a systematic literature review on the following topics:

Computational Thinking, online learning, post-primary students and teaching/learning.

Chapter 4 outlines the philosophical and methodological framework that underpins the study. The rationale for this PhD study's use of pragmatism and EDR are presented. The methodological method and analytical techniques are discussed in detail.

The prototype phase commences with Version 1 of the Computational Thinking course. This consists of a teacher handbook, slides, lesson plans, videos and unplugged activities instructions. In these products laid the first iteration of design principles which would evolve to become the ADAPTTER Framework. During this phase, two versions of the course were evaluated and subsequently revised.

Chapter 5 provides an overview of the initial content and design of the Computational Thinking course. It clarifies what is understood by the terms: practical, engagement, quality, effectiveness, and low threshold.

Chapter 6 concerns the piloting of Version 1 of the Computational Thinking course. It presents a detailed narrative on its implementation and evaluation. This version was piloted in four schools with eighty-six students and five teachers.

Chapter 7 concerns the piloting of Version 2 of the Computational Thinking course. This pilot was concerned with the practicality and sustainability of the design and course. This version was piloted in two schools with 340 students and six teachers.

Semi-Summative Phase

At the end of an EDR study, a summative evaluation of the intervention is performed to ascertain if the intervention can be determined effective. In reality, this phase is considered a semi-summative phase, as it usually includes recommendations for future versions.

Chapter 8 concerns the semi-summative evaluation and trial of the Computational Thinking course and corresponding instructional design. This version was piloted in two schools with 20 students and one teacher.

Chapter 9 is concerned with the contributions of this PhD study. The principal contributions of this research is a Computational Thinking course, and the ADAPTTTER framework, which can both be used by teachers and students when learning, teaching and designing Computational Thinking courses. These contributions have implications for educational policy, initial teacher education, second-level curriculum, and second-level teaching practice in Ireland. The use of unplugged activities for teaching Computational Thinking was shown to be a strength by both teachers and students. How Computational Thinking is defined in the Irish media, policy, educational and curriculum documents was investigated and a definition that caters to the Irish position was proposed.

2 What is Computational Thinking?

‘I don’t know’ (Student C12, pre-test)

‘Something to do with computers + thinking’ (Student C9, pre-test)

2.1 Introduction

Computational Thinking is at the core of this PhD, specifically the designing, developing, and evaluating of a Computational Thinking course for Irish second-level students.

Unfortunately, it suffers from definitional confusion. Its exact meaning is frequently debated. This has implications as how Computational Thinking is defined and understood in policy and curriculum documents affects how it is taught in schools. This chapter starts with first reviewing how Computational Thinking is defined in academic literature and also literature connected with educational organisations. It next explores how Computational Thinking is defined in Irish educational policy documents and how this definition impacts the Irish compulsory education curriculum. Finally, Irish newspapers are reviewed to ascertain how Computational Thinking is understood in the public consciousness. This chapter ends with stating the definition of Computational Thinking used in this PhD study. The rationale for its use is provided from the findings from the review of academic, Irish educational, policy, and media documents.

2.2 Computational Thinking: The Genesis

What is Computational Thinking? There is no simple answer to this question. A lack of consensus exists among academics regarding a universally accepted definition. These differences were highlighted in 2009 at ‘The Scope and Nature of Computational Thinking’ workshop, where participants failed to reach a consensus concerning the content and structure of Computational Thinking (National Research Council, 2010).

The term has its beginning with a Danish concept *datalogisk tænkning*, (Computer Science Thinking) to describe the analysis of data representations, processing and media (Naur

1970 as cited in Caeli and Yadav, 2020). Although it is Seymour Papert (1980) and his much cited book *'Mindstorms: Children, Computers, and Powerful Ideas'* that is credited with its beginning. In his introduction, Papert states:

I shall be talking about how computers may affect the way people *think and learn*. I begin to characterize my perspective by noting a distinction between two ways *computers might enhance thinking and change patterns of access to knowledge* [emphasis added] (Papert, 1980, p. 3)

Further into his book he makes the first usage of the term Computational Thinking in print, this is in connection with the mental skills children develop when programming (Papert, 1980, p. 182). Twenty-six years later, Jeannette Wing, in her seminal 2006 viewpoint article, popularised more widely the term 'Computational Thinking'. She defined it as follows 'Computational thinking involves solving problems, designing systems, and understanding human behavior, by drawing on the concepts fundamental to Computer Science' (Wing, 2006, p. 33). In this same influential 2006 article, she highlighted that Computational Thinking is a fundamental skill that is applicable to all. She emphasised the following points: it is concerned with conceptualising not programming; composed of mathematical and engineering thinking: concerned with ideas not artefacts, with how humans think not computers (Wing, 2006). Over the following five years, she twice refined her original definition (Wing, 2008, 2011) but never digressed from her original viewpoint that it is fundamentally a problem-solving framework.

Computational thinking is the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent (Cuny *et al.*, 2010 (unpublished) cited in (Wing, 2011).

This final and oft-cited definition of Wing's has significance for compulsory education in that it states that Computational Thinking is a thought process. It is independent of technology, and its solutions are executed by Computational Thinking agents that can be considered either a human, a computer, or both.

2.3 Computational Thinking: Problem Solving

Wing's definitions (2006, 2008, 2011) do not stand alone, many others exist, such as Aho (2011), Barr and Stephenson (2011), The Royal Society (Furber, 2012), Computing at School (CAS) (2015) Bocconi *et al.* (2016), Shute *et al.* (2017) and Denning (2017).

Aho simplified Wing's definition by stating how it can be considered as a 'thought processes involved in formulating problems so that their solutions can be represented as computational steps and algorithms' (2011, p. 832). This definition, while deceptively simple, does highlight the word computational, and in doing so shares similarities with the Royal Society's 2012 definition:

Computational thinking is the process of recognising aspects of computation in the world that surrounds us, and applying tools and techniques from Computer Science to understand and reason about both natural and artificial systems and processes (Furber, 2012, p. 29)

Denning (2017) is in favour of the Aho (2011) definition as it refers to a computational model. He articulates strongly that a Computational Thinking definition should make mention of a computational model, as it is through engagement with this model that we achieve our tasks (i.e. solve our problems). He also strongly articulates that an algorithm is not an arbitrary series of steps and should not be described as such but as steps that are under the control of a computational model, they do not require human judgement. Hence he does not consider a human as a Computational Thinking agent.

Barr and Stephenson (2011) also consider Computational Thinking as a problem-solving framework, but their definition is concerned with the operational side of Computational Thinking. Their definition forms part of the International Society for Technology in Education (ISTE) and the Computer Science Teachers Association (CSTA) definition of Computational Thinking (ISTE and CSTA, 2011a). Both the CSTA and ISTE wanted a definition that educators could use to build and develop Computational Thinking across the

K-12 curriculum irrespective of the subject (ISTE and CSTA, 2011b). Their definition is as follows:

Computational thinking (CT) is a problem-solving process that includes (but is not limited to) the following characteristics:

Formulating problems in a way that enables us to use a computer and other tools to help solve them.

Logically organizing and analyzing data.

Representing data through abstractions such as models and simulations.

Automating solutions through algorithmic thinking (a series of ordered steps).

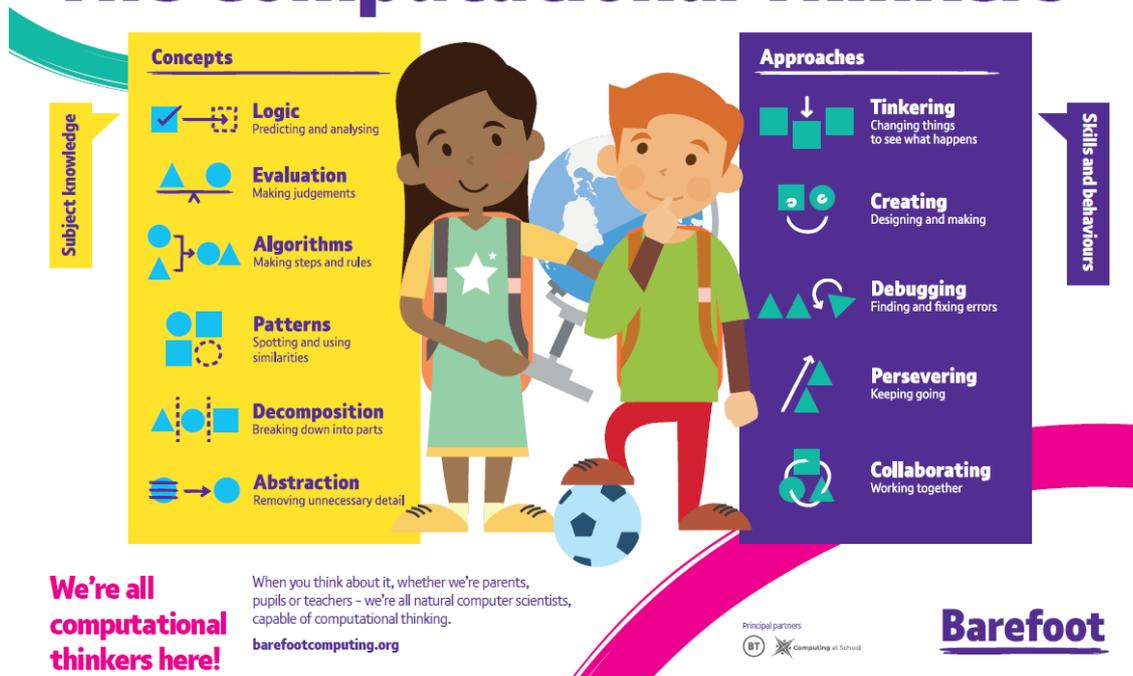
Identifying, analyzing, and implementing possible solutions with the goal of achieving the most efficient and effective combination of steps and resources.

Generalizing and transferring this problem-solving process to a wide variety of problems. (ISTE and CSTA, 2011a, para. 2)

The necessary attitudes that were essential to Computational Thinking were also listed.

The ISTE and CSTA definition shares many viewpoints with that of the Barefoot Computing At School (CAS) project, a UK project concerned with teaching at the compulsory education level. Barefoot CAS supports UK teachers in delivering the computing curriculum at primary level (Barefoot CAS, 2014). The Barefoot CAS definition too, is operational in nature. It involves six concepts (logic, algorithms, decomposition, patterns, abstraction and evaluation) and five approaches (tinkering, creating, debugging, preserving and collaborating). Like Wing's definition, they too see Computational Thinking as a problem-solving process that is independent of technology. The Computational Thinking is done before the computer is involved. However, the problem is specified in a way that the computer can both understand and solve (but the 'thinking' is done before this technology is introduced).

The Computational Thinkers



**Figure 2-1 Barefoot Computing diagram of Computational Thinking (Barefoot 2020)
Reproduced with permission**

Similar to the above approaches are the definitions provided by both Bocconi *et al.* (2016) and Shute *et al.* (2017). Both their definitions arose from literature reviews conducted to establish the most popular definitions and characteristics of Computational Thinking. Bocconi *et al.*'s (2016) definition resulted from juxtaposing the Computational Thinking concepts and skills found in seminal papers (Barr and Stephenson, 2011; Lee *et al.*, 2011; Grover and Pea, 2013; Selby and Woollard, 2014a; Angeli *et al.*, 2016) with Wing's 2011 definition.

CT describes the thought process entailed in formulating a problem so as to admit a computational solution involving abstraction, algorithmic thinking, automation, decomposition, debugging and generalisation. (Bocconi *et al.*, 2016, p. 17)

Bocconi *et al.* use the word 'automation' in their definition, which implies the use of a computer. Wing documents how 'Computing is the automation of our abstraction' (2008, p. 3718), and while a computer is needed to interpret these abstractions, she highlights how a human can be considered a computer. Wing (2008) also highlights the combination of a human computer and a machine computer, as a way to exploit the processing power of

humans. The definition from Shute, Sun and Asbell-Clarke (2017) shares parallels with that from Bocconi *et al.* (2016): ‘A conceptual foundation required to solve problems effectively and efficiently (i.e. algorithmically with or without the assistance of computers) with solutions that are reusable in different contexts’ (Shute, Sun and Asbell-Clarke, 2017, p. 151). Shute, Sun and Asbell-Clarke (2017) also categorised Computational Thinking into six main concepts: decomposition, abstraction, algorithms, debugging, iteration and generalisation. It is evident from the literature that a subset of concepts associated with Computational Thinking appear in the literature, and a working definition for this PhD’s CT course should include them.

2.4 Computational Thinking: Programming Languages

What is noteworthy about the previous definitions is that programming languages are not explicitly mentioned. Wing’s 2006 paper and her subsequent definitions (2008, 2011) are very particular about this. Her emphasis is clear: Computational Thinking is about conceptualising, not programming. It is a thought process that is concerned with ideas, not artefacts. This is not a universally held belief.

At the aforementioned ‘The Scope and Nature of Computational Thinking’ workshop some participants were not of the same opinion. Ursula Wolz, Mitchel Resnick, Roy Pea, and Eric Roberts all voiced opinions concurring that programming is essential to Computational Thinking (National Research Council, 2010):

as soon as we think about the origins of computational thinking and computational literacies, programming has been at the heartland of the definition and the abstractions that are created as step-by-step algorithmic procedures. (Roy Pea quoted in National Research Council, 2010, p. 13)

In fact, Brennan and Resnik (2013) proposed a definition of Computational Thinking centred on the Scratch programming language, and Ioannidou *et al.*’s (2011) description focused on computational patterns and how they can be used in the development of artefacts like games and simulations.

While it is evident that a single, agreed definition for Computational Thinking does not exist, some core concepts and skills are evident in the aforementioned definitions. They are as follows: abstraction, algorithmic thinking, decomposition, pattern matching and evaluation (Barr and Stephenson, 2011; Bocconi *et al.*, 2016; Computing at School (CAS), 2015; Shute *et al.*, 2017). Nevertheless, the place of programming in the definition of Computational Thinking must be addressed; if programming is considered essential to Computational Thinking then this greatly affects curriculum design, and subsequently whether its application is broad or narrow. Wing (2006) argued that Computational Thinking was applicable outside the subject of Computer Science. The Royal Society (Great Britain) (2017) highlight how computing underpins many disciplines such as design, social science, arts and mathematics; thus, it follows that its problem-solving framework can be applied across disciplines. One can also argue that its problem-solving methodology is relevant in other subjects, with or without programming (Lu and Fletcher, 2009; Barr and Stephenson, 2011; Yadav *et al.*, 2014). Barr and Stephenson's (2011) seminal paper 'Bring Computational Thinking to K12' provides examples of how Computational Thinking elements can apply across different subject domains. The above points regarding core concepts, programming and Computational Thinking being relevant outside of Computer Science were key facts in establishing a definition of CT for this PhD's CT course (see 2.7).

2.5 Computational Thinking in Irish Policy documents

To ascertain Irish policymakers conceptualisation and scope of Computational Thinking, a document analysis was conducted of Irish educational documents (Oct 2018 to Feb 2019). This involved three databases: the Irish Department of Education and Skills publication and media library, the National Council for Curriculum and Assessment (NCCA) publication library, and the Irish Curriculum online site. One document was selected

through pearl growing from the publications available on the Creative Ireland site. Pearl growing is where one document is used to find other relevant documents. This resulted in an initial selection of 248 documents.

Table 2-1 Irish Policy Database Information and no. of documents returned from each database

Database	Url	Details	Number
Irish Department of Education and Skills publication and media library	https://www.education.ie/en/Publications/Policy-Reports/	Policy Reports Folder	116
Irish Department of Education and Skills publication and media library	https://www.education.ie/en/Publications/Education-Reports/	Education Reports Folder	129
National Council for Curriculum and Assessment publication library	https://www.ncca.ie/en/publications-and-research/publications	Search Used 'Computational Thinking'	8
Irish Curriculum online	https://www.curriculumonline.ie/	Search Used 'Computational Thinking'	2
Creative Ireland Publications	https://www.creativeireland.gov.ie/en/publications/	Selected through pearl growing: document titled 'Creative Youth'	1

The first pass filtered the articles based on the title. This resulted in thirty-seven documents being included in the second pass. These documents were scanned to determine if they should be included in the study. This resulted in nine documents being classified as relevant. Their titles are listed as follows:

- Creative Youth: A plan to enable the creative potential of every child and young person.
- Digital Strategy for Schools 2015-2020: Enhancing Teaching, Learning and Assessment.
- Digital Strategy for Schools 2015-2020 Action Plan 2018.
- Senior Cycle Key Skills Framework.
- Key Skills of Junior Cycle Managing Information and Thinking.
- Short Course Coding Specification for Junior Cycle.

- Computer Science Curriculum Specification Leaving Certificate Ordinary and Higher Level.
- Background Paper and Brief for the development of a new Primary Mathematics Curriculum.
- Primary Mathematics Curriculum Draft Specification Junior Infants to Second Class for Consultation.

2.5.1 Definitions

A definition of Computational Thinking was provided in two documents: the background document for the primary maths curriculum and the Computer Science Leaving Certificate curriculum. Wing's (2011) influence is evident in both definitions. In the primary maths background document (2016), it is defined as a thought process used to solve complex problems that are executed by a human or a machine. It also mentions core concepts and operational characteristics such as ‘logical reasoning (predicting and analysing), algorithms (devising steps and rules), decomposition (breaking down a problem into parts), patterns and generalisations (identifying and using simulations), abstractions (removing unnecessary detail) and evaluation (making judgements)’ (NCCA, 2016, p. 27). Unlike Wing’s (2011) definition the Leaving Certificate specification (NCCA and DES, 2018) makes mention of technology, stating that this ‘problem-solving methodology’ ‘involves the implementation of solutions using automation, programming and computer systems’ (NCCA and DES, 2018, pp. 6, 7).

2.5.2 Applicability

What is interesting about the background primary maths document and the draft primary mathematics curriculum document is that they refer to the broad applicability of Computational Thinking (for example across subjects such as English and Science) but with the underlying goal of students learning to code (NCCA, 2016; NCCA and Primary Developments, 2018). The draft primary mathematics curriculum outlines how Computational Thinking is being introduced with the purpose of forming a basis for coding. It aims to ‘embed the basis of coding— Computational thinking, and flexible and

creative thinking skills— in the new curriculum specification for mathematics currently under development’ (NCCA and Primary Developments, 2018, p. 71). It further outlines how they intend to explore different approaches of integrating coding into the primary curriculum (NCCA and Primary Developments, 2018).

This focus on coding is also evident at the post-primary level. At second-level, Computational Thinking is only referenced in the Leaving Certificate Computer Science and Junior Cycle Coding Short Course (NCCA and DES, 2016, 2018). However, the problem-solving and thinking skills associated with Computer Science are recognised outside this discipline in the Leaving Certificate Computer Science specification. Coupled with this fact, the Senior Cycle Key Skills framework document identifies five key skills essential to teaching and learning at the post-primary senior cycle level (NCCA, 2009). Information processing and critical and creative thinking are two such skills (NCCA, 2009). The critical and creative thinking key skill is further developed to outline its sub-components. They are as follows: examining patterns and relationships, identifying and analysing problems and solutions, and recording, organising, summarising and integrating information. These sub-skills share parallels with Computational Thinking approaches. In fact, the Computer Science Curriculum document draws one’s attention to these parallels in its discussion of critical and creative thinking. At the Junior Cycle level, ‘managing information and thinking’ is an identified key skill (NCCA, 2015). This information is important as it helps to answer the following question: is Computational Thinking only valuable to Irish students in the capacity that it will facilitate them to be programmers or is the problem-solving equally as important? Whilst the Irish educational documents highlighted the importance of Computational Thinking in connection to coding, Irish educational documents also highlight components of CT in the Senior Cycle Key Skills framework document and also the Leaving Certificate Computer Science specification (NCCA, 2009; NCCA and DES, 2018).

2.6 Computational Thinking in the Irish media

Irish media publications were analysed to ascertain the view of Computational Thinking in the public consciousness. This review formed part of EDR's preliminary analysis (see 4.4.3). To this end, the Nexis database was searched using the term 'Computational Thinking' (March 2017) and filtered to Irish Newspapers. It returned fifty-one documents, the earliest being dated as 2011. Duplicates were removed resulting in forty-five documents.

The screenshot shows the Nexis database search results for the query "Computational Thinking". The interface includes a navigation bar with "LexisNexis Services", "What's New", "English (UK)", "My Documents (0)", "Contact Us", "Help", and "Sign Out". Below the navigation bar, there are tabs for "Search" and "Sources". The search results are displayed in a table format, sorted by "Newest to Oldest". The results are filtered to "Newspapers" (48), "Web-based Publications" (2), and "Magazines & Journals" (1). The table shows three results:

Rank	Title	Source	Words	Hits	Date	Preview
1.	Minister launches Tech Week 2017	Business World (Digest) TECHNOLOGY, Robert McHugh	310 words	1 hit	30 March 2017	Preview
2.	Counting on school pupils to hit higher standards in maths and reading skills	Irish Independent NEWS; Pg. 25, Katherine Donnelly	462 words	1 hit	15 March 2017	Preview
3.	Government promises to implement 400 actions from education plan this year	BreakingNews.ie IRISH NEWS,	774 words	1 hit	06 February 2017	Preview

Figure 2-2 The Nexis Database

A close reading of each article in the corpus was made to 1) understand how the articles conceptualised Computational Thinking, 2) ascertain the most common mentioned pedagogies, and 3) ascertain if the articles linked to policy documents.

A summary of the initial findings is now made. The media articles as a whole can be classified as descriptive, nearly all written in response to an event such as the Hour of Code initiative, a competition (Bebras), a conference or promotion of a new higher-level educational course or maker space. Only five articles defined Computational Thinking, with one explicitly stating Computational Thinking is concerned with programming (Sunday Business Post, 2012). The other definitions saw broader applicability of Computational Thinking such as its connection to problem-solving, understanding human

behaviour and communicating information processes (Boran, 2013; Maguire and Power, 2015; Quinlan, 2016). For the articles that do not provide a definition, contextual and morphemic cues were used (Nagy and Scott, 2000) to establish one. This resulted in surmising that the word Computational Thinking was synonymous with either Computer Science (14 articles) or coding (14 articles) see Figure 2-3. Whilst Computational Thinking is not programming, many articles do not make this distinction.

But a crucial "language" of the 21st Century all too often is overlooked or not even considered -- the language of computer programming. It is clear that day by day the ability to engage in what is loftily referred to as computational thinking is becoming more and more important. (Harris, 2012, para. 1,2)

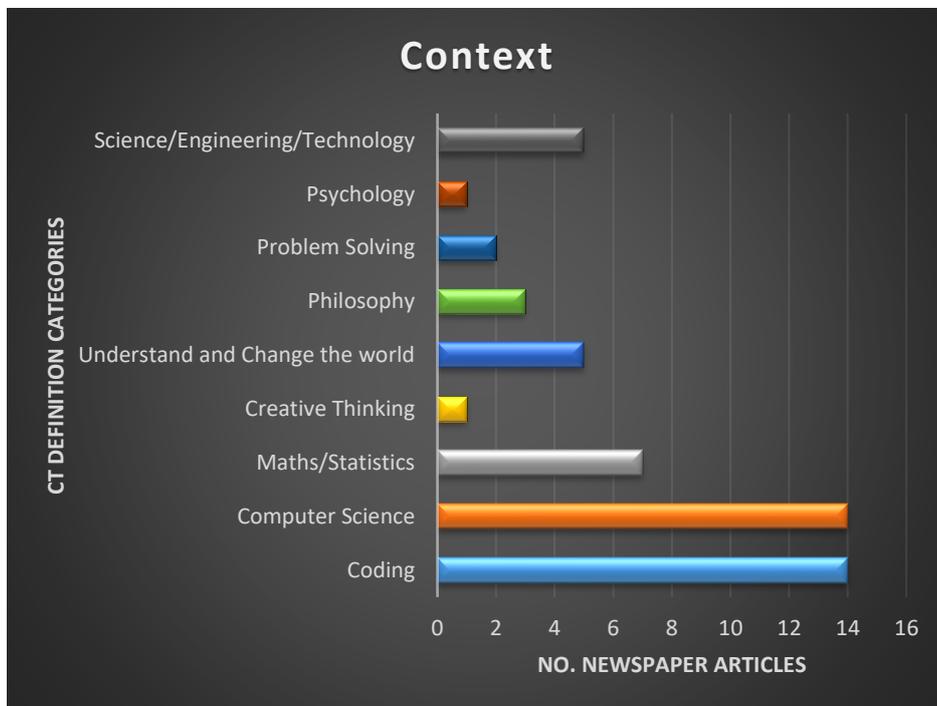


Figure 2-3 Contextual and Morphemic cues used to surmise Computational Thinking definition

Seven policy documents were referenced in the articles, with one being from the UK. This point is important as it highlights the need for Irish Policy documents to contain a consistent definition of Computational Thinking.

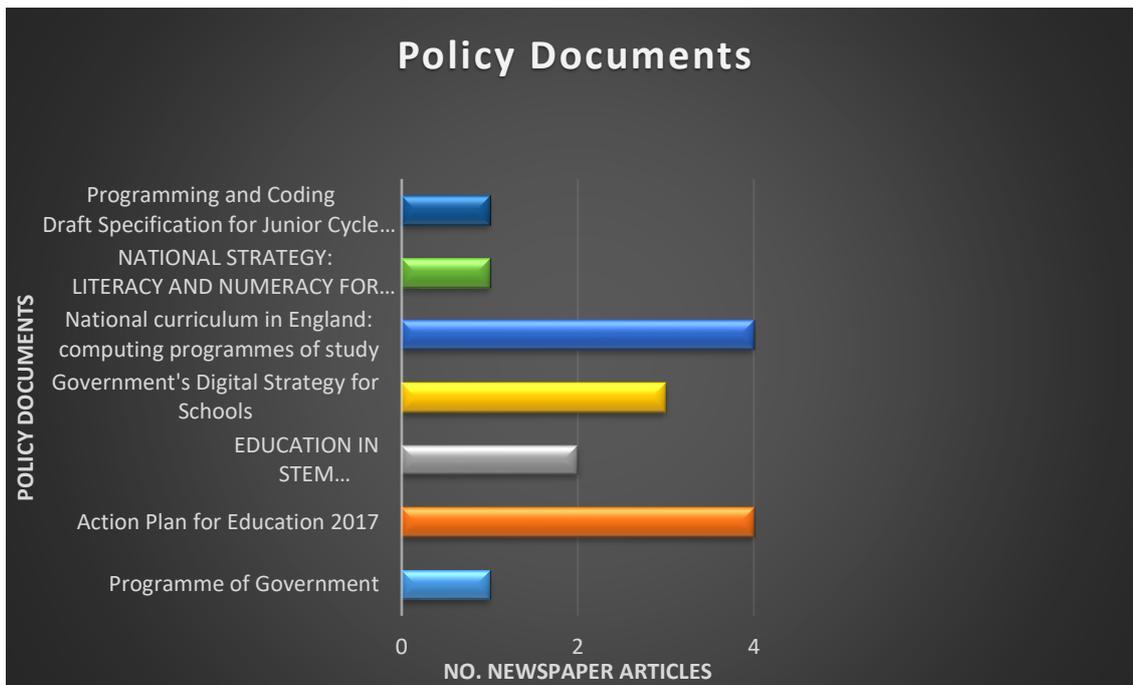


Figure 2-4 Policy Documents mentioned in Irish Media

2.7 Computational Thinking: Working towards a definition

Definitions are important, they facilitate pedagogy, curriculum design and assessment (Selby and Woollard, 2014a; Kirwan, Costello and Donlon, 2018). A definition of Computational Thinking for this PhD’s CT course is needed and needs to consider academic literature, educational organisations, and Irish policy/educational documents.

This chapter opened with a question: what is Computational Thinking? No simple answer was forthcoming, but what is evident from the analysis of the academic and educational literature is that a subset of concepts recursively appears in the literature. They are as follows: abstraction, algorithmic thinking, decomposition, pattern matching and evaluation (Barr and Stephenson, 2011; Computing at School (CAS), 2015; Bocconi *et al.*, 2016; Shute, Sun and Asbell-Clarke, 2017). It is also apparent that differences surrounding the scope and applicability of Computational Thinking exist. They concern the exact meaning of a computational agent (human, machine or both) and the extent of the applicability of

Computational Thinking (relevance outside of the discipline of Computer Science)
(Curzon *et al.*, 2019).

The Irish policy documents highlighted the importance of Computational Thinking in connection to coding whilst also simultaneously highlighting the problem-solving framework of Computer Science. Curzon *et al* (2019) pose the following question: if only a machine can be considered a computational agent, then Computational Thinking may be viewed as only concerning programming. Hence why then do we need the term Computational Thinking, surely programming is the skill?

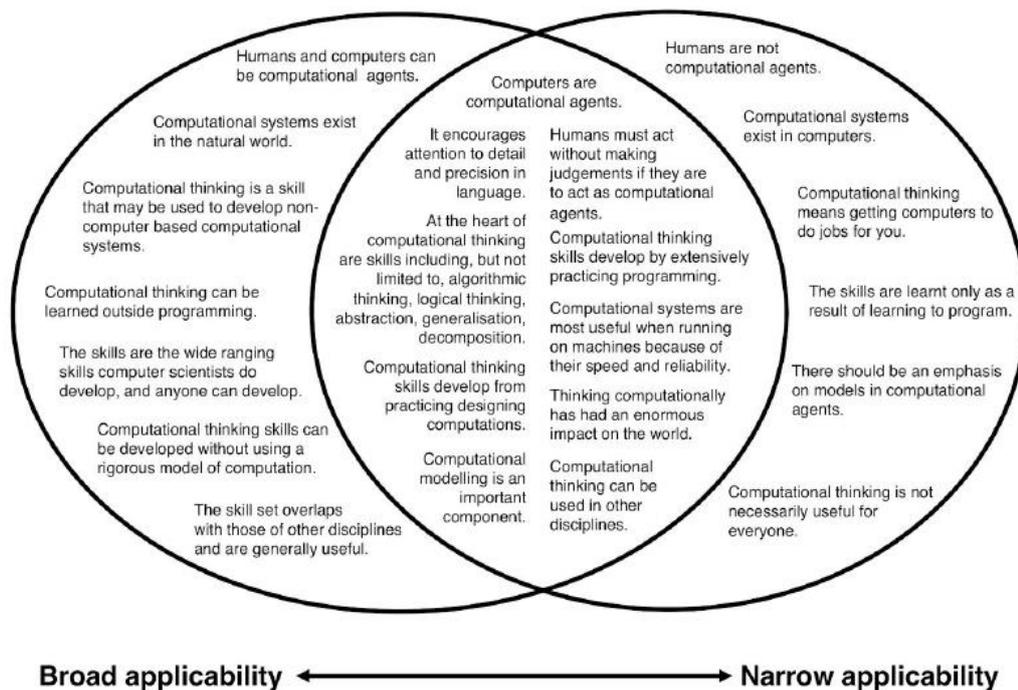


Figure 2-5 ‘Agreement and disagreement around two views of what Computational Thinking should be.’ (Curzon *et al.*, 2019, p. 515) Copyright The Cambridge University Press. Reproduced with permission of The Cambridge University Press through PLSclear.

Curzon *et al.* (2019) diagram (see Figure 2-5) shows the broad and narrow applicability of Computational Thinking. With reference to this diagram, this PhD study proposes a centralised view. It recognises the importance of the core concepts of Computational

Thinking. However, it sees Computational Thinking agents as being both human and machine, thus recognising the value of the problem-solving framework outside Computer Science. This PhD study uses the definition from Selby and Woollard

computational thinking is a brain-based activity that enables problems to be resolved, situations better understood, and values better expressed through systematic application of abstraction, decomposition, algorithmic design, generalisation, and evaluation in the production of an automation implementable by a digital or human computing device. (Selby and Woollard, 2014, p. 20)

This definition was chosen, 1) as it incorporates operational characteristics, 2) shares parallels with organizations concerned with the teaching at the compulsory education level, 3) shares core concepts and skills with other definitions of Computational Thinking and 4) shares Wing's (2011) viewpoint of a problem-solving framework. The operational characteristics such as abstraction, decomposition etc., are explained in Chapter 4.

2.8 Conclusion

This chapter explored the genesis, background and scope of the term Computational Thinking. It investigated how it was defined in academic literature and Irish policy and media documents. A definition of Computational Thinking that caters to the broad applicability of Computational Thinking whilst recognising its importance to Computer Science was presented. In the next chapter, a systematic literature review is conducted to ascertain the role programming plays in the teaching of Computational Thinking. Parts of the systematic literature review were previously published (Kirwan, Costello and Donlon, 2018).

3 Systematic Literature Review

3.1 Introduction

This chapter presents the results of a systematic literature review conducted as a due diligence process into the role of programming languages in the learning and teaching of Computational Thinking online. This review was timebound and conducted in the early stages of this PhD study, when it was envisaged that the proposed solution would be available online. It captured the wide range of tools and pedagogical approaches used in the teaching of Computational Thinking. While the online part is currently outside the scope of this PhD study, the findings for this review directly influenced the Computational Thinking Course design and the final ADAPTTER model. A systematic approach to the literature review was conducted to provide a transparent and accountable methodology that can be replicated (Gough, Oliver and Thomas, 2012).

3.2 Methodology

The following section provides an overview of the methodology used to identify, analyse and assess the relevant literature on the aforementioned topics. With this aim in mind, many texts were consulted, including Gough *et al.* (2012), Rutter *et al.* (2013), Kitchenham and Charters (2007) and EPPI-Centre (2006). While all provide a broad agreement on what constitute the main stages of a systematic review, Kitchenham and Charter's (2007) process was the primary influencer on this research. This was for two reasons: it contained a well-documented process and is aimed at PhD students. The following documented steps from Kitchenham and Charters process (2007) are listed as sub-headings in the next section 1) Specification of the Research Question, 2) Development of a Review Protocol, 3) Identification of research 4) Selection of primary studies, 5) Study Quality assessment, 6) Data extraction and monitoring.

The following tools were also consulted: the Prisma flowchart to illustrate the flow of information through the above systematic stages (Moher *et al.*, 2009) and the EPPI centre

guidelines on 'Extracting data and quality assessing primary studies in Educational Research' (EPPI-Centre, 2003) for the purpose of creating a google data extraction form.

3.2.1 Specification of the Research Question

To guide the specifications of the literature review questions and the search string strategy, the PICO (Population, Intervention, Comparison and Outcome) framework was used (Tianjing and Dickersin, 2016):

Population: K12 children aged 12-18

Intervention: Teaching of Computational Thinking online without using a programming language

Comparison: Teaching of Computational Thinking online using a programming language

Outcome: Effectiveness and extensiveness of the teaching medium and the effectiveness of the pedagogical approach.

The intention was to ensure the body of reviewed literature provided answers to the review questions as a whole instead of providing multiple answers (Gough, Oliver and Thomas, 2012). The returned literature would be concerned with the different approaches used to teach and learn Computational Thinking online. The filling out of the PICO framework resulted in the following set of questions.

With respect to second-level students age 12-18 years:

What are the current pedagogical approaches to teaching Computational Thinking online?

The following Sub-section Review Question (SRQ) was also devised:

- How is computational thinking defined in the selected studies?

3.2.2 Development of a Review Protocol

A pilot study was conducted using the Compendex and Inspec databases to first develop and then fine-tune the review instruments. The outcome of this process was:

- The documentation of the search strings to be used against the selected digital libraries (see 3.2.3).
- The inclusion/exclusion criteria for the literature selection see (see 3.2.5, 3.2.6).
- A list of the data to be extracted from the analysed documents.

All of the above is discussed in the following sections.

3.2.3 Identification of Research

A search strategy was developed. It aimed to choose search terms that would strike a balance between specificity (finding relevant articles) and sensitivity (finding all articles on a subject) (EPPI Centre, 2006). The keywords ('K12 students', 'Computational Thinking', 'Online', and 'Teaching') were provided from the PICO framework. These were used to generate synonyms that were Boolean Or(ed) together (see Table 3-1). For example, the term algorithmic thinking was used as a synonym for Computational Thinking, as it can be considered a sub-component of Computational Thinking (Bocconi *et al.*, 2016). This process resulted in constructing a search string based on linking the four identified OR(ed) synonym lists using the word AND. However, the aforementioned pilot study resulted in the refactoring of this search string. The 'K12 Students' search terms were removed to broaden the search criteria and return more literature.

Thus the final search string was constructed by ANDing the three OR lists consisting of synonyms for 'Computational Thinking', 'teaching', and 'online', which was then run against six databases. Note search strings were subtly changed based on the databases used. For example, the synonym 'logical thinking' was removed from the Education Databases search due to the number of false positives that arose.

Table 3-1 Search string composed of synonyms used in the systematic literature review

Keyword	Synonyms
K12 Students	K-12 students (age 12-18) (((“Year 7” OR “Years 7” OR “Year 8” OR “Years 8” OR “Year 9” OR “Years 9” OR “Year 10” OR “Years 10” OR “Year 11” OR “Years 11” OR “Year 12” OR “Years 12” OR “grade 7” OR “grades 7” OR “seventh grade” OR “grade 8” OR “grades 8” OR “eighth grade” OR “grade 9” OR “grades 9” OR “ninth grade” OR “grade 10” OR “grades 10” OR “tenth grade” OR “grade 11” OR “grades 11” OR “eleventh grade” OR “grade 12” OR “grades 12” OR “twelfth grade” OR “grammar school” OR “grammar schools” OR “middle school” OR “middle schools” OR “six form” OR “sixth form” OR “comprehensive schools” OR “comprehensive school” OR “secondary” OR “high school” OR “high schools” OR “prep school” OR “prep schools” OR “college preparatory school” OR “college preparatory schools” OR “senior school” OR “senior schools” OR “pre-college” OR pre near/1 college OR “K12” OR “junior high school” OR “junior high schools” OR “Junior Cycle” OR “Senior Cycle” OR “Transition Year”))
Online	Online OR on_line OR MOOC OR “Massive Open Online Course” OR “distance education” OR “web-based course” OR blended
Teaching	Teach* OR education OR learn*
Computational Thinking	(“logical thinking” and comput*) OR “computational thinking” OR “algorithmic thinking”.

3.2.4 Selection of primary studies

The databases/libraries used were Inspec, Compendex, ACM Digital Library, Education Research Complete, ERIC and British Education Index. These libraries were chosen as they are from both fields concerned with this PhD research, Computer Science and education, and for the number of journals they contain. In total 800 articles were returned and subjected to the study selection process.

Table 3-2 Academic Database Information and no. of articles returned from each database

Database	Characteristics	Articles Returned
INSPEC	11+ million journal articles, 4.5+ conference papers	137
Compendex	11.6 million journal articles’ 6.9 million conference papers	
ACM Digital Library	2.7 million articles	68
Education Research Complete	1,300 education journals’ 2,300 conference papers	550

ERIC	1.6 million records (including over 630,000 full-text documents)	42
British Education Index	238,000 articles	3

As recommended by the Kitchenham and Charters' (2007) guidelines, inclusion and exclusion criteria were developed. This resulted in the abstract and title of each of the aforementioned 800 articles being screened based on the following criteria:

3.2.5 Inclusion Criteria

- Articles are available in English
- Articles concerned with teaching or learning of Computational Thinking online
- Online was defined based on where the teaching occurred or whether the tool was self-contained
- Articles that correlated to the definitions of Online, Blended, Flexible or Web Enhanced as defined by the Online Learning Consortium (OLC) were included (Sener, 2015)
- Empirical research
- Journal or conference articles

3.2.6 Exclusion Criteria

- Conceptual/opinion pieces
- Duplicates

The outcome of this process was that 331 articles were passed to the second stage of study selection. This resulted in the BibTeX references for these files being imported into a library named Merge LitReview in the Qiqqa reference management software. Whereby each reference was then associated with the following word tags '1st Pass' and an abbreviation of their digital source platform, for example, Ebsco or ACM.

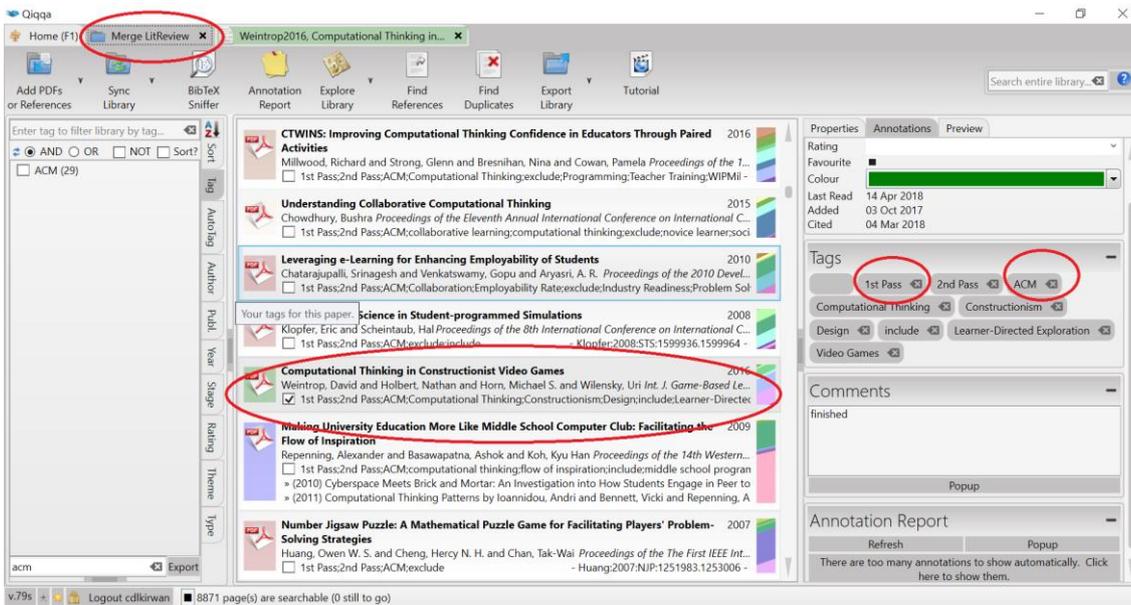


Figure 3-1 Qiqqa Reference Tool

During the second stage, the full article for each reference was retrieved. This resulted in the methodology and conclusion section being read in detail and once again screened against the inclusion/exclusion criteria. This resulted in forty-six articles being passed to the final literature review. Forty-four of the included articles were empirical studies (Two papers included as part of the pearl process were not. These two articles were included as they added pertinent information to existing studies, and were written by the same original study authors.)

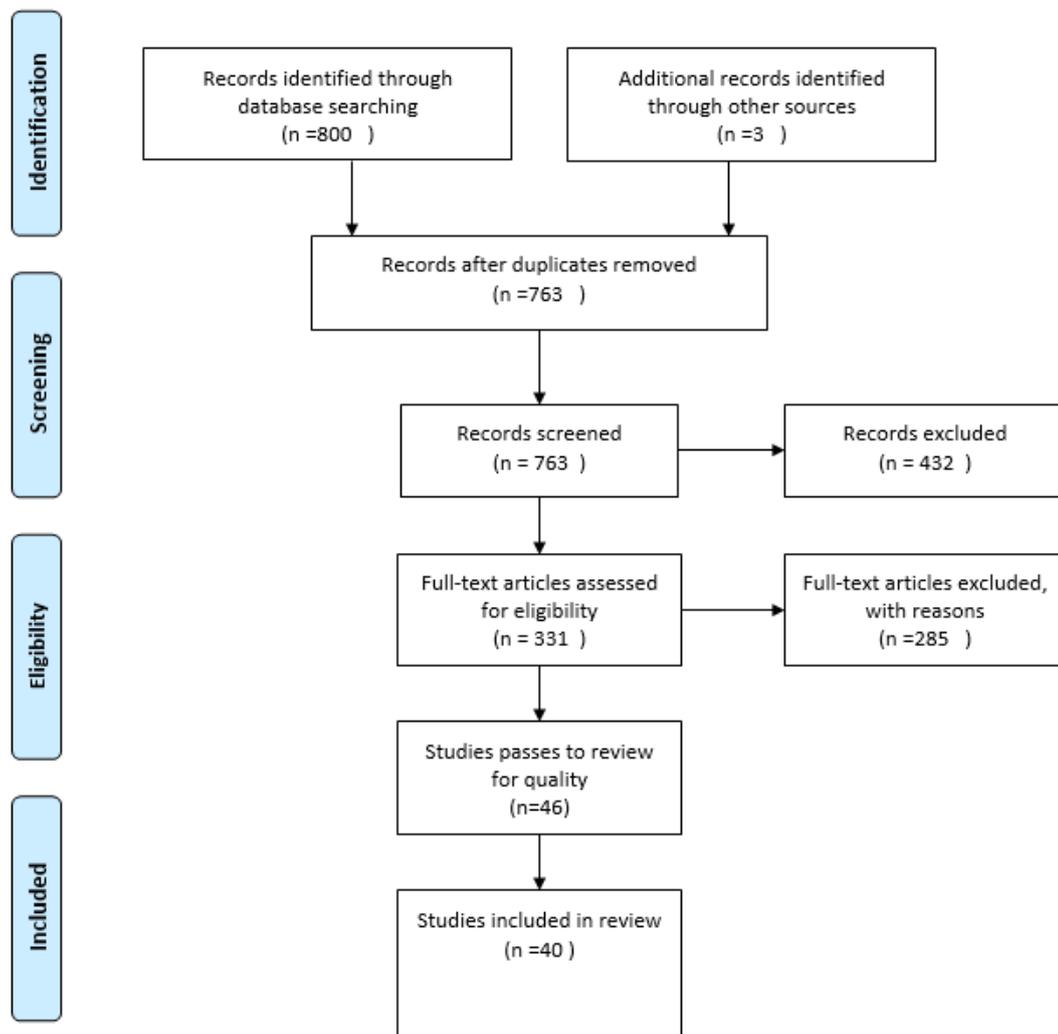


Figure 3-2 Study selection process (Moher et al., 2009). Original printed (Kirwan, Costello and Donlon, 2018)

3.2.7 Study Quality Assessment

Kitchenham and Charters (2007) strongly recommend that all selected articles be screened for quality in addition to the inclusion/exclusion criteria. With this end in mind, the final forty-six papers were screened against the five prompts provided by Dixon-Woods *et al.* (2006) that assist with making judgments on the quality of a paper. The prompts are as follows:

- Are the aims and objectives of the research clearly stated?
- Is the research design clearly specified and appropriate for the aims and objectives of the research?

- Do the researchers provide a clear account of the process by which their findings were reproduced?
- Do the researchers display enough data to support their interpretations and conclusions?
- Is the method of analysis appropriate and adequately explicated? (Dixon-Woods *et al.*, 2006, p. 4)

Six articles were subsequently removed due to their descriptive nature and lack of empirical data.

3.2.8 Data extraction and monitoring

A data extraction form was developed using a Google form. The data extraction form aimed to assist with the accuracy of recording details from the selected primary studies. It was initially piloted against the Compendex and Inspec database. JavaScript code was also written and attached to this form to edit individual responses in case of errors or add new data questions that arose during this pilot of the data collection phase. The collected data included all the answers necessary to answer the literature review questions and the above quality assessment criteria. The EPPI-Centre (2003) 'Guidelines for extracting data and quality assessing primary studies in educational research' provided an excellent resource in aiding the development of the extraction questions.

Figure 3-3 A screenshot of a snippet of the Google extraction form used to record information about the reviewed articles

3.3 Literature Review

This section provides a review of the forty papers selected based on the aforementioned systematic literature review process. These papers were examined to provide answers to the following questions.

1. What are the current pedagogical approaches and tools to teaching Computational Thinking online?
2. How is Computational Thinking defined in the articles?

The reviewed papers were grouped into categories based on the tools typically used to teach Computational Thinking. These categories were Visual Programming Languages (video game design), Playing Video Games, Programming Languages (excluding visual), Unplugged Activities, Competitions, Collaboration Tools and Other. To facilitate comparisons, categories are presented and discussed using the same headings. These headings are: ‘Tools’, ‘Theories and Techniques’ and ‘Notable Points’. Where necessary to aid comprehension, academic literature beyond the reviewed forty papers were included in the discussion.

3.3.1 What are the current pedagogical approaches to teaching Computational Thinking online?

3.3.1.1 Visual Programming Language

Table 3-3 List of articles that use visual programming languages to teach Computational Thinking

Number	Article	Tool
1	Ahmadi and Jazayeri (2014)	AgentWeb
2	Ahmadi, Jazayeri and Landoni (2012)	AgentWeb
3	Basawapatna (2016)	Simulation Creation Kit
4	Dasgupta, S. <i>et al.</i> (2016)	Scratch,
5	Koh <i>et al.</i> (2014)	Assessment
6	Marshall (2011)	Assessment
7	Escherle <i>et al.</i> (2016)	AgentCube
8	Repenning <i>et al.</i> (2009)	AgentSheets
9	Marcelino <i>et al.</i> (2017)	Scratch
10	Grover, Pea (2015)	Scratch
11	Grover, Pea (2016)	Scratch
12	Basawapatna and Repenning (2010)	AgentSheets
13	Snodgrass (2016)	Code.org and also Scratch
14	Jenkins (2015)	Poem generator (Uses a scratch extension)
15	Xie and Abelson (2016)	App Inventor
16	Berland and Wilensky (2015)	VBOT (graphical programming language)
17	Basawapatna <i>et al.</i> (2011)	Agent Sheets

Visual programming languages was the most popular learning approach, appearing in seventeen papers. A core purpose of visual programming environments is to teach programming languages to novice students. The reasoning is that this environment bypass (or reduce) the syntactic difficulties associated with computer programming languages (Kelleher and Pausch, 2005). Therefore a shift in practice now exists in that they are being used to teach Computational Thinking (Ahmadi and Jazayeri, 2014).

3.3.1.1.1 Tools:

The visual programming languages described in the above seventeen papers, can be reduced to just two languages: Scratch and Agent modelling (AgentSheet derivatives). Scratch is a block-based visual programming language that is primarily targeted at children and is used to create interactive stories, games and animation (Lifelong Kindergarten Group MIT Media Lab, no date). AgentSheets is a drag and drop programming language. It facilitates the creation of agent-based games and simulations (AgentSheets 2014).

3.3.1.1.2 Learning Frameworks and Techniques

The learning theory most commonly associated with visual programming is constructionism, as the learning results in the building of an artefact, for example, a game (Ahmadi, Jazayeri and Landoni, 2012; Marcelino *et al.*, 2017). This relationship also proved true in this PhD study, with constructionism being the most popular approach observed in the seventeen game design papers (Ahmadi, Jazayeri and Landoni, 2012; Pellas and Peroutseas, 2016). This relationship between constructionism and the visual programming languages (Scratch and AgentSheets) was expected, but what was unexpected was the triad connection that existed between visual programming languages, the pedagogical approach and the definition of Computational Thinking. This connection was evident in the majority of game design papers.

The papers concerned with AgentSheet derivatives revolved around a definition of Computational Thinking that incorporated Computational Thinking Patterns (Marshall, 2011; Ahmadi, Jazayeri and Landoni, 2012; Koh *et al.*, 2014; Basawapatna, 2016). In contrast, most of the papers concerned with the Scratch environment revolved around Brennan and Resnick's (2012) definition of Computational Thinking. This concerned the mapping of Scratch blocks to Computational Thinking concepts, for example, conditionals, loops, and sequences (Dasgupta *et al.*, 2016; Xie and Abelson, 2016; Marcelino *et al.*, 2017). Both these definitions were core to the teaching and assessment of Computational

Thinking in these studies. This fact highlights an important issue concerning how Computational Thinking is taught, specifically when one considers its transferability and utility. Jenkins (2015) articulates this issue by questioning how synonymous Computational Thinking is with problem-solving when you are assessing it by programming constructs and concepts such as for example loops and sequences.

Whilst constructionism was the main learning theory observed, it was not the only one: Zone of Proximal framework (Basawapatna *et al.*, 2013; Koh *et al.*, 2014; Escherle *et al.*, 2016), Foundations for Advanced Computational Thinking framework (FACT) (Grover, Pea and Cooper, 2015) and Five Flow of Inspiration Principles (Repenning, Basawapatna and Koh, 2009)(Repenning *et al.* 2009) frameworks were also adhered to. Of particular mention is the Zone of Proximal Flow Framework, as it is concerned with engagement.

3.3.1.1.2.1 Zone of Proximal Flow Framework

Zone of Proximal Flow Framework is a combination of two pedagogical concepts Csikszentmihalyi's theory of flow and Vygotsky's Zone of Proximal Development (ZPD). Csikszentmihalyi's flow theory states that a person is both completely engaged and motivated with an activity when their obtained skill set matches the challenges they are undertaking. Vygotsky's theory is concerned with 'Zone of Proximal Development' which involves describing the difference between what a student can do, with and without guidance (Basawapatna *et al.*, 2013)

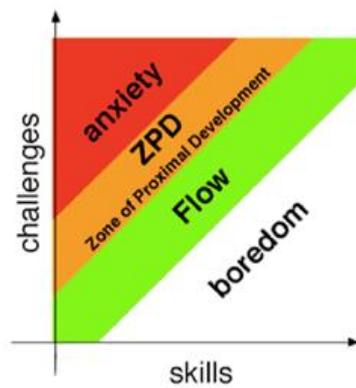


Figure 3-4 Zones of Proximal Flow diagram where ZPD is located between regions of flow and anxiety ‘Republished with permission of ACM, from Basawapatna et al., 2013; permission conveyed through Copyright Clearance Center, Inc’.

The ZPD is located between regions of flow and anxiety (see Figure 3-4) (Basawapatna et al., 2013). Students were guided through their game design challenges using a gentle-slope approach. They first developed a simple game, and only when mastery was achieved did they progress to more complex patterns. If the student’s skill set was unable for the programming challenge, guidance was provided in the form of class instruction, online tutorials, and peer learning among classmates (Basawapatna et al., 2013).

3.3.1.1.3 Other Techniques: Collaboration and Remixing in Video Game Design

An important teaching approach detected in many of the game design studies in this review was student collaboration (Basawapatna and Repenning, 2010; Ahmadi and Jazayeri, 2014; Marcelino et al., 2017). An approach facilitated by the online environment as it supported both synchronous and asynchronous communication. This is particularly observed in the studies by Repenning et al. (2009) and Basawapatna and Repenning (2010), where an online homework submission system called the Scalable Game Design Arcade (SGDA) was employed. Repenning et al.’s (2009) study document how this Scalable Game Design Arcade (SGDA) was devised to reproduce the interactions of a Middle School computer club. Programming projects were displayed online in a public forum to be both viewed and run by students. Students could then provide feedback on said projects and also download the code. Whilst the projects were created using AgentSheets, peer to peer teaching (and

interactions) were facilitated indirectly by the SGDA environment. By viewing and running each others' code, students learned from each other.

Another teaching approach used to learn Computational Thinking was remixing. Three articles studied this approach, two informally concerning the aforementioned SCDA studies, and one directly. Using an impressive dataset consisting of just under two and a quarter million projects, with 173,000 users, Dasgupta *et al.* (2016) investigated if programmers increased their programmatic skills and learned Computational Thinking by remixing other programmer's code. Using quantitative analysis, their findings supported this theory.

3.3.1.1.4 Notable Points

Visual programming languages were the most popular intervention identified in this review for teaching Computational Thinking online. This approach is not without its flaws. Two pertinent points put forward by Basawapatna *et al.* (2011) and Ahmadi *et al.* (2012) need to be considered. Basawapatna *et al.* (2011) highlight how there now exists in some studies the implicit belief that just by teaching programming, one learns Computational Thinking. Ahmadi *et al.* (2012) put forward a similar argument stating that game design tools do not inherently or explicitly support Computational Thinking. It is introduced through the teacher's guidance. These points are important when one considers that many teachers need to be upskilled to teach Computational Thinking. In general, game design environments lack specific support for directly teaching Computational Thinking. Novice students learn how to use the tools, but not how to apply Computational Thinking strategies (Ahmadi *et al.* 2012). To help overcome this issue, Ahmadi *et al.* (2012) recommend the use of targeted videos inside the game design environment.

3.3.1.2 *Playing Video Games*

Table 3-4 List of articles that document playing a video game to teach Computational Thinking

Number	Author	Tool
1	Holbert and Wilensky (2011)	Formula T Racing
2	Weintrop <i>et al.</i> (2016)	Formula T Racing. RoboBuilder:- Program to Play
3	Wood (1980)	Mastermind
4	Dhatsuwan and Precharattana (2016)	Blockyland
5	Guenaga <i>et al.</i> (2017)	Make World
6	Pella and Peroutseas (2016)	SecondLife with Scratch4SL

Section 3.3.1.1 detailed how visual programming languages were used to teach Computational Thinking. This was achieved through the construction of artefacts, such as video games. This section is concerned with the playing of video games. Seven studies revealed (see Table 3-4) how simply playing a video game can be a successful way of practising and in turn learning Computational Thinking. However, two of the seven studies did incorporate game design with game playing (Pellas and Peroutseas, 2016; Weintrop *et al.*, 2016). Nevertheless, a dependency between the teaching tool, pedagogical approaches, and the Computational Thinking definition was once again observed. For example, Holbert and Wilensky (2011) defined Computational Thinking as the ‘ability to translate or encode phenomena (real or imagined) into representations that leverage computational power’ (2011, p. 110). Their way of teaching Computational Thinking incorporated a theory known as computational encoding.

3.3.1.2.1 Learning Frameworks and Techniques

Constructionism was the most dominant theory observed due to ‘constructionist’ activities being core to the gameplay. For example, Make World is concerned with making worlds to teach STEAM concepts, Blockyland is a city building game, and FormulaT Racing (FTR) is a racing game, where players control the car’s speed by painting sections of the race track. Other teaching approaches observed were collaboration and immersion (Wood,

1980; Dhatsuwan and Precharattana, 2016; Pellas and Peroutseas, 2016; Guenaga *et al.*, 2017), which are described in the next section.

3.3.1.2.2 Collaboration

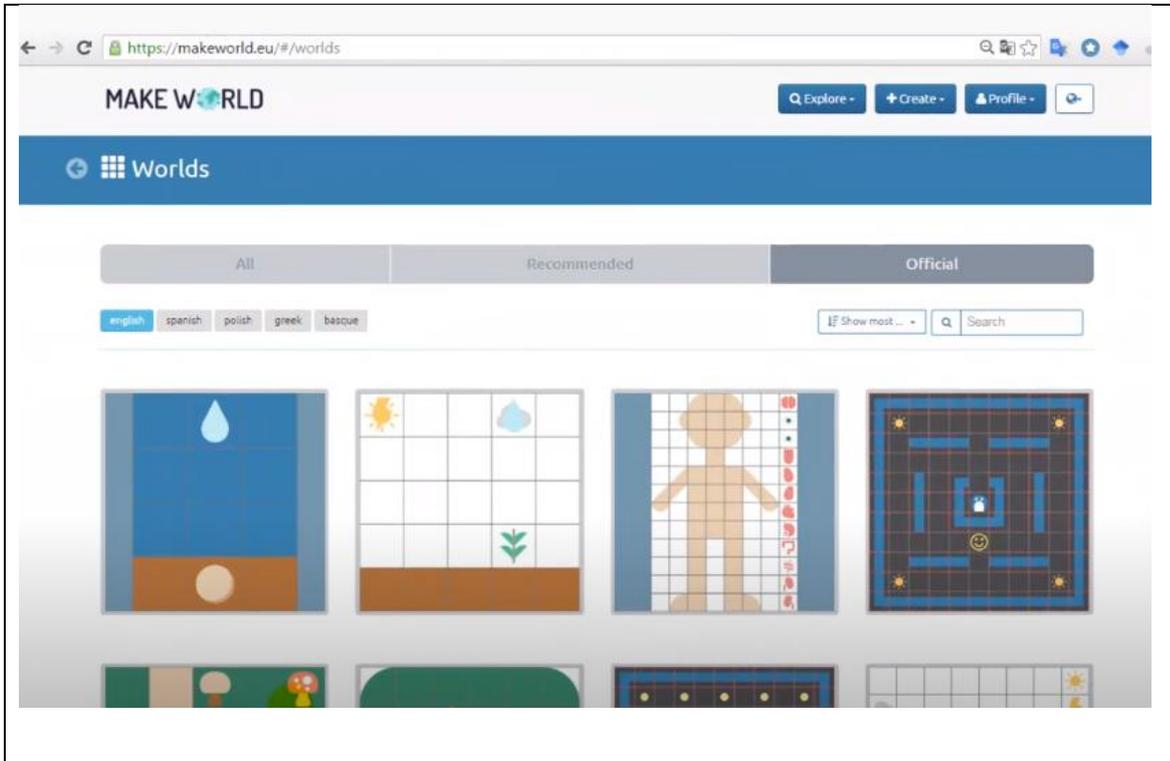


Figure 3-5 A screen capture of the MakeWorld website at makeworld.eu

Collaboration is a key feature of the video games Make World and Blockyland (Dhatsuwan and Precharattana, 2016; Guenaga *et al.*, 2017). In particular, the MakeWorld platform was designed to teach science by using Computational Thinking, thus developing STEAM skills. The site provides ready-to-use worlds and stories, enabling users to remix existing resources when creating their new worlds. Collaboration is enabled by users following and commenting on other players' activities.

3.3.1.2.3 Immersion

Futschek (2006) posits that algorithmic thinking can be learned independently of a programming language. Debabi and Bensebaa (2016) build on this theory, they proposed the AlgoGame which allows participants to focus on problem-solving without having to

learn the syntax of a programming language first. They reported that after playing the AlgoGame, an experimental group had better results in writing a selection sort algorithm than a control group. In their study, AlgoGame was very much the ‘teaching tool’. It was developed to embrace immersion and real-life situations.

3.3.1.2.4 Notable Points

The playing video games category was a surprising find. It highlighted how Computational Thinking could be taught by playing specific constructivist based games. This has relevance when considered in relation to the self-directed learning of Computational Thinking and also as a precursor to learning programming. Wood’s (1980) study was concerned with a self-contained tool used to research logical thinking and constructive feedback. The tool used in Wood’s study was a computer program developed based on the Mastermind game. Mastermind is a two-person game where you have a code maker and a codebreaker, the objective being to break the code. In Wood (1980) study the code maker was the computer. The purpose of his study was to observe if corrective feedback to the user (i.e. codebreaker) during the game (for example, in instances where the code guesses are not consistent with known information) would improve a person’s logical thinking skills. With this aim, a standardised logical thinking test was administered pre and post intervention. His study demonstrated a statistically reliable improvement in logical thinking for the experimental group, indicating that logical skills are transient (Wood, 1980). This paper was included in the literature review as the Mastermind game is now available online, and Strom and Barlo (2011) provide background information on the logical reasoning behind this tool.

3.3.1.3 Programming languages (not visual-based)

Table 3-5 List of articles that use non-visual programming languages to teach Computational Thinking

Article	Tool
Krugel and Hubwieser (2017)	Object-Oriented programming, mixture of web tools: SVG.JS library, CindyJS , Blockly-Games, UmpleOnline, Java-Tutor and Codeboard
Grandell (2005)	no specific language mentioned

Two studies (see Table 3-5) analysed the teaching of Computational Thinking using tools/languages that do not fall specifically/exclusively into the game design category or visual-based programming languages. Krugel and Hubwieser’s (2017) study was concerned with learning object programming using Computational Thinking as a springboard. Grandell’s (2005) study is concerned with introducing university-level Computer Science content to high school students. Constructivism was the most dominant theory observed in both studies. However, Grandell’s (2005) course was also influenced by a new conceptual model known as ActiWe (Active on the Web), where the guiding principle is active learning.

3.3.1.3.1 Notable Points

Krugel and Hubwieser (2017) highlighted an interesting didactical dilemma regarding the teaching of Object-Oriented Programming. They advocated teaching in a ‘real-life context’, but to do that with reference to programming one must learn a lot of complex concepts quickly. To overcome this difficulty, they initially hid advanced topics, by introducing students to Computational Thinking and object-oriented concepts first before programming, to reduce the cognitive load.

3.3.1.4 Un-Plugged Activities

Table 3-6 List of articles that use un-plugged activities to teach Computational Thinking

Number	Article	Tools
1	Miller <i>et al.</i> 2013	computational creative exercises (Those adapted for K-12 are available on the Google Exploring Computational Thinking website.)
2	Shell <i>et al.</i> 2014	computational creative exercises
3	Vivian <i>et al.</i> 2014	Unplugged activities and Visual Programming https://www.edx.org/course/think-create-code-adelaidex-code101x-3 https://csermoocs.adelaide.edu.au/moocs/

Three articles looked at activities that teach Computational Thinking using activities that do not require a computer. Two of these articles employed computational creative exercises (Miller *et al.*, 2013; Shell *et al.*, 2014). The goals of these studies were to investigate if the learning of Computational Thinking can be improved if it is combined with creative thinking. These exercises were all hands-on and required problems to be solved collaboratively using written analysis and reflection. A web-based wiki system (Written Agora system) was employed to facilitate the collaboration.

The third study was concerned with teaching Computational Thinking using a mixed approach, i.e. unplugged and programming (Vivian, Falkner and Falkner, 2014). Visual programming was introduced after Computational Thinking concepts were understood. Vivian, Falkner and Falkner (2014) study was predominantly concerned with the theory behind the development of a Mass Open Online Course for teaching Computer Science.

3.3.1.4.1 Notable Points

Miller *et al.*'s (2013) study was successful in its aims. Their research demonstrated that Computational Thinking and course achievement increased with each completed exercise.

However, despite this success, students were dissatisfied with the exercises. They did not believe they were effective in preparing them to use computational and Computer Science tools in their chosen fields. They also had difficulties collaborating with students outside of class. Shell *et al.* (2014) study took on board these findings, with their exercises being more aligned with course topics and emphasised more in class.

3.3.1.5 Competitions

Table 3-7 List of articles that use competitions to teach Computational Thinking

Number	Article	Tool
1	Dagiene and Stupuriene 2016	Bebras Competition
2	Izu <i>et al.</i> 2017	Bebras Competition

Two articles referred to the Bebras competitions (Dagiene and Stupuriene, 2016; Izu *et al.*, 2017). The Bebras competition was initially set up to introduce information technology and informatics to students. Its purpose has now changed, becoming an approach for developing Computational Thinking and deeper learning of informatics. The competitions can be considered informal learning-by-doing, as students engage in solving both engaging and challenging tasks (Haberman *et al.*, 2011). They can also be used as an innovative way of assessing Computational Thinking (Dagiene and Stupuriene, 2016). Dagiene and Stupuriene (2016) study was a systematic literature review consisting of fifty-six papers. Of note in their findings are the pedagogical approaches associated with Bebras: concept-based learning and task-based assessment. Their study concluded by stating that there is evidence that Bebras is an important tool for promoting Computational Thinking and problem-solving Izu *et al.* (2017, p. 42) study classified Bebras tasks in terms of the Computational Thinking categories defined by Barendsen *et al.* (2015) ‘data collection (DC), data analysis (DA), data representation (DR), problem decomposition, abstraction, algorithms & procedures, automation, parallelization and simulation’. Their finding highlighted that algorithms and data representations were the Computational Thinking

concepts most prevalent in the competition, accounting for 75–90% of the tasks. They also state that concepts central to Computational Thinking are alien to many K12 teachers.

3.3.1.6 Collaboration Tool

Table 3-8 List of Collaboration Tools

Number	Article	Tool
1	Othman <i>et al.</i> (2015)	Online Collaborative Learning System
2	Wilkerson-Jerde (2014)	Categorizer

Two particular studies formed their own category as they were specifically about how the tool is used to develop Computational Thinking using collaboration. The study of Othman *et al.* (2015) investigated cognitive enhancement in introductory programming by using an Online Collaborative Learning System. The intervention used a ‘Think-Pair-Share’ approach, i.e. a collective cortex method with the aim of enhancing logical thinking. Wilkerson-Jerde (2014) study also focused on collaborative environments. Her tool, called the Categorizer, allowed 11-14-year-old students to construct, share and categorise computational artefacts that they developed based on fractals. Her study facilitated collaborative inquiry while building on the theory of constructionism, in that shared artefacts were built, but the patterns and themes of these fractals were explored as a collaborative effort by a community.

3.3.1.7 Other

Table 3-9 List of articles that form their own category

Number	Article	Tool
1	Tsai and Tsai 2017	Excel
2	Tsai <i>et al.</i> 2017	Excel
3	Mouza <i>et al.</i> 2017	Unplugged and scratch
4	Liao and Liang (2017)	Learning approaches
5	Korkmaz <i>et al.</i> (2017)	Assessment scales

6	Bremgartner, Netto and Menezes, 2017	ArCare
---	--------------------------------------	--------

Six papers fitted into this last category. Two discussed the teaching of Computational Thinking using the application software Excel (Tsai and Tsai, 2017; Tsai *et al.*, 2017) with mixed results. One study focused specifically on what pre-service teachers need to know to teach Computational Thinking outside Computer Science, i.e. in different disciplines, for example, mathematics, science and literacy (Mouza *et al.*, 2017). Liao and Liang's (2017) study focused on learning approaches. Their study sought to research if blended learning can promote Computational Thinking. Finally, the study by Korkmaz, Cakir and Özden (2017) was concerned with assessment and Computational Thinking Scales. Bremgartner *et al.*'s (2017) study gives very little detail on how Computational Thinking was taught in their blended course, but does give information on their online environment called ArCARE (Conceptual Framework of Educational Resources Adaptation in VLEs).

3.3.1.7.1 Notable Points

A notable point is that once again, the importance of the Computational Thinking definition was evident. In both Tsai *et al.* (2017) studies, Computational Thinking was defined based on Excel using the definition by Yeh *et al.* (2017).

3.3.2 Literature Review Findings

The findings from this literature review directly influenced the practical outcomes of this research: the Computational Thinking course and the final ADAPTTTER model. The findings also served to support the rationale for not using programming languages, and to investigate the use of unplugged activities as the main teaching tool.

A summary of the literature review is as follows: programming languages were the most common tool used to teach Computational Thinking, appearing in over half of the papers. This is consistent with the findings from other literature reviews (Lockwood, 2019),

concerned with teaching Computational Thinking. Visual programming languages were the most popular tool observed, however these tools were developed to teach programming languages. They were not designed to teach Computational Thinking; this is introduced through the guidance and support of the teacher (Ahmadi, Jazayeri and Landoni, 2012). Teachers and subsequently students first have to learn a programming language, albeit a low floor language, in conjunction with learning Computational Thinking. This reduces the low-threshold component of the course.

A significant finding from this review is the connection between the definition of Computational Thinking (CT), the teaching tool, and the assessment. This PhD study takes a centralised view in connection with the applicability of CT (see 2.7). It proposes a definition that highlights that Computational Thinking is a thought process and is applicable outside programming whilst still recognising its importance in Computer Science. The articles in this PhD study highlighted the connection observed in papers between the teaching tool and programming constructs such as loops, sequence, conditionals which would narrow the applicability of the course. Related to this point, is the didactical dilemma highlighted by Krugel and Hubwieser (2017) and discussed in the literature review regarding the teaching of Object Oriented Programming. They encourage teaching in a ‘real-life context’, but recognise that this is very difficult with programming, as one would have to learn many difficult concepts quickly. With reference to teaching Computational Thinking, if the problems undertaken are to be based in the real world, the solution to a problem would be dependent on students’ programming abilities. Vivian *et al.* (2014) study highlighted that Computer Science is more than programming and that it is important to teach concepts and computational thinking prior to visual programming. All of the above points support the view of using unplugged activities, as they can be used to tackle complex problems and topics without the hurdle of learning to program first (Bell *et al.*, 2009).

An interesting finding from the review was the use of playing games as a strategy for teaching Computational Thinking. Debabi and Bensebaa (2016) study built on Futschek (2006) theory that algorithmic thinking can be learned independently of a programming language. Whilst Wood (1980) study showed how logical thinking could be improved by playing the Codebreaker game. A version of the Codebreaker game will be included in the Computational Thinking course, to support the development of logical thinking which is an important part of CT (Selby *et al.*, 2013; Curzon *et al.*, 2019). Similar to Krugel and Hubwieser (2017), Debabi and Bensebaa's (2016) study highlighted the use of real-life situations. This point was also addressed indirectly in Miller *et al.*'s, (2013) study, where students were dissatisfied with the use of unplugged exercises. They did not believe they were effective in preparing them to use computational and Computer Science tools in their chosen fields. This PhD's CT course will ensure students see the relevance of each skill they learn, and its relationship to Computer Science.

3.3.2.1 Pedagogical Approaches

The most common theories observed in the literature were constructionism, Csikszentmihalyi's theory of flow and Vygotsky's zone of Proximal Development . Constructionism is very much related to programming as the teaching is concerned with producing an artefact. Csikszentmihalyi's theory of flow highlights the importance of engagement in ensuring content and activities are pitched at the right level for students. This is an important point when one considers that Izu *et al.* (2017) stated that concepts central to Computational Thinking concepts were alien to many K12 teachers. The FACT approach devised by Grover and Pea advocated the view that no singular pedagogical approach facilitate their study's goals of deeper learning. The most common pedagogical approaches observed in this PhD study were collaboration, scaffolding and remixing.

3.4 Conclusion

This chapter confirmed the view that teaching Computational Thinking using programming was not compatible with the goals of this PhD's proposed course. Two of these goals (which are discussed in Chapter 5 (5.3.1 and 5.4.1) were low threshold and real-world situations. An approach was needed to allow both teachers and students to tackle real-world problems (with a connection to Computer Science) in a way that was not limited by programming knowledge. Although programming did form part of the course, it was not the main teaching tool. Unplugged activities were posited as such an approach, and their usage was investigated in this PhD study. However, as only five articles used this approach in this literature review, a further review of the literature was needed, see Chapter 5. The next chapter presents an overview of the methodological framework of the course, that of EDR, which was used to design a Computational Course and instructional framework.

4 Methodology: Educational Design Research

I herewith ask all young scientists to renounce the false modesty of previous generations of scientists. Do not be afraid to name the agent of the action in a sentence even when it is “I” or “we”. (Day, 1995, p. 166)

4.1 Introduction

This chapter provides an overview of this study's philosophical underpinning, pragmatism, and its relation to this PhD study's methodological framework, EDR. A justification for EDR is provided. The chapter then outlines the rationale for methods and analytical techniques used in this study. A comprehensive account is provided to avoid the repetition of methodological detail in future chapters when results are discussed. Finally, the chapter concludes with a discussion on the role of ethics in this study.

The chapter is written in the first person. The reason for this is I, as the sole researcher, wanted to detail and justify my methodological decisions without the ‘positivistic detachment of the classical sociologist’ (Starfield and Ravelli, 2006, p. 222). I am also aware that it is my beliefs on knowledge, that form the worldview that underpins this research.

4.2 Worldview: Pragmatism

Pragmatism is a philosophical approach that originated in America in the 1890s. Charles Sanders Peirce, William James and John Dewey are credited with being its founders (Legg and Hookway, 2019). At its core, pragmatism incorporates many ideas, it values both subjective and objective knowledge. It is concerned with what works and finding solutions to problems. It can use diverse approaches and is inherently linked to action (Hall, 2013; Creswell, 2014). Pragmatism sees reality as being an ever-changing process that is not static (Maxcy, 2003). It is this worldview that found immediate resonance with me and underpins my research. This is in no small way due to my background.

4.2.1 **Pragmatism Rationale**

I have a diverse background, with experience in the fields of medicine, software engineering and education. A positivist worldview surrounded my medical expertise. I qualified as a therapeutic radiographer in a teaching hospital that undertook many clinical trials. My software engineering experience was immersed in the practical, specifically evolutionary prototyping, a methodology that parallels design research (Anderson and Shattuck, 2012). I worked as a Senior Software Engineer for eleven years. I experienced the benefits of developing multiple prototypes to investigate and then solve complex problems. I experienced the advantage of learning by doing. My current and past educational experiences demonstrate the benefit of obtaining multiple perspectives to help understand and answer research questions (Winstone *et al.*, 2017). I have worked in academia for the last nine years and understand the importance of engaging in feedback dialogue with my students. I thus value both subjective and objective knowledge and see the benefit of fitting the method to the problem (Creswell and Plano Clark, 2011).

4.2.2 **Dewey: Theory of Knowledge and Inquiry**

It is the pragmatic philosophy of John Dewey that appeals to me, precisely his problem-solving approach to knowledge inquiry and his view on teaching, that you learn by doing (both very prominent features of this PhD) (Dewey, 1938a, 1938b). A pertinent question about knowledge acquisition is how the mind can obtain knowledge about the world outside itself (Biesta, 2010)? This question challenged Dewey (1911), he calls it the inevitable question, how is knowledge possible when one considers that the knower ‘is purely individual or “subjective”... whose being is wholly psychical and immaterial and a world .. which is purely universal or “objective” and whose being is wholly mechanical and physical’ (1911, p. 441).

Dewey answers this question by using a theory that is not founded on the aforementioned dualistic mind-world framework but on transactional realism (Dewey, 1929a, 1929b;

Biesta, 2010). He proposes that the mind and world are in continuous interactions with each other through transactions. Knowledge is obtained through these actions as individuals engage with and change their environment. Thus, in simple terms, to know the world you have to interact with it, you will only know the world in relation to your actions, results, and reflections on said actions, i.e. you do not achieve a once for all truth. It is not separated from a lived life (Dewey, 1929a; Biesta, 2010).

For Dewey, knowledge or what he called ‘warranted assertions’ was the outcome of inquiries that relied on both actions and reflections (Dewey, 1938b, p. 9). These warranted assertions should be shared with all, with the aim to improve society (Hall, 2013). To explain further, Dewey’s philosophy of knowledge is dependent on his concept of inquiry (Dewey, 1910, 1938b) (depicted in Figure 4-1), where he believed that inquiry began from a problem-solving conception. This results in the gathering of evidence and data to solve problems (Morgan, 2014). This theory corresponds with Design Research (see section 4.4) which uses mixed methods (Brown, 1992), as the evidence and data used to solve a problem can be either qualitative or quantitative. For Dewey, different knowledge is just the result of the different actions used to interact with the world (Biesta, 2010).

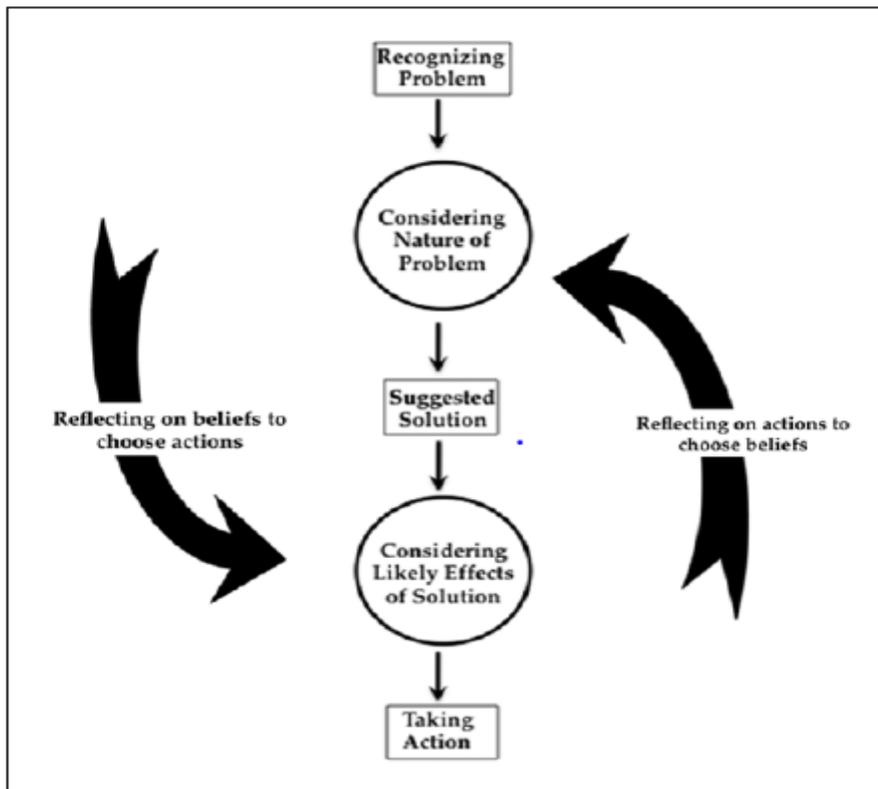


Figure 4-1 Dewey's model of inquiry Morgan (2014) (Reprinted with permission)

4.3 Research Approach

At its core, this study is about the development and design of an intervention, a Computational Thinking course. This intervention is developed to be engaging, high quality, effective, and low-threshold, (i.e. minimum background computer knowledge is needed).

While designing, developing and assessing this PhD study's course, the effectiveness of unplugged activities as a pedagogy for teaching Computational Thinking is evaluated. Coupled with this is an evaluation of the merit of the design: how practical and effective it is for its users. The study's overarching research question guides all these aims.

What are the characteristics of a practical, high quality, engaging, effective, and low threshold course for learning and teaching Computational Thinking to Irish second-level teachers and students?

4.3.1 Educational Design Research

EDR is the methodological framework used in this study. It can be defined as:

the systematic study of designing, developing and evaluating educational interventions (such as programs, teaching-learning strategies and materials, products and systems) as solutions for complex problems in educational practice, which also aims at advancing our knowledge about the characteristics of these interventions and the processes of designing and developing them. (Plomp, 2013, p. 13)

It is important to note that ‘design research’ can be considered as a common label assigned to ‘a family of related research’ (Van den Akker *et al.* (2006, p. 4). Examples include design experiments (Collins, 1990; Brown, 1992), design-based research (The Design-Based Research Collective, 2003) and design studies (Shavelson *et al.*, 2003).

EDR is characterised as the following:

- Interventionist: the intervention occurs in the real world
- Iterative: it includes a cyclic approach of design, evaluation and revision
- Process-oriented: the emphasis is on improving and understanding the intervention
- Utility oriented: practical, the merit of the design is measured
- Theory-oriented: some or all the design is based upon theoretical propositions.
- The end design contributes to theory building (Van den Akker *et al.*, 2006).

As a methodology, it is distinguished from other forms of inquiry in that it focuses on developing practical solutions and theoretical insights simultaneously (Plomp, 2010).

Reeves’s (2006) summary diagram (see Figure 4-2) highlights the primary milestones/approaches in design research, i.e. 1) analysis, 2) prototype development and 3) reflection.

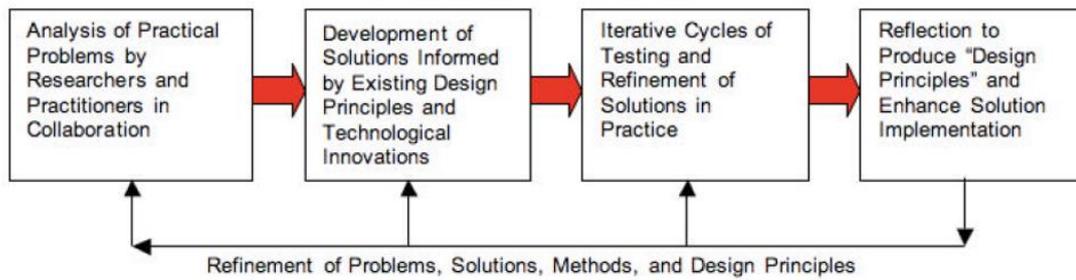


Figure 4-2 Reeves (2006) Design research approaches in educational technology research (Reproduced with permission of The Licensor through PLSclear.)

4.3.2 Motivation and Justification for EDR

As a pragmatist, I value both subjective and objective knowledge. I see the benefit of fitting the method to the problem (Creswell and Plano Clark, 2011). The problem is how to design and evaluate a Computational Thinking course for second-level students and their teachers.

EDR is recommended when:

- Content knowledge is new
- Instructional materials, and teachers' knowledge is poor
- Teaching and pedagogical expertise are unclear (to both teachers and researchers)
- Multifaceted policy, societal or political factors may negatively affect progress (Kelly, 2013)

All of the above factors are relevant to the defined problem and research question that this study addresses. These factors are now discussed under the headings of Researcher Experience, Teacher Experience, Irish Policy documents and Computational Thinking Content, and School Setup.

4.3.2.1 Researcher Experience

I have a wealth of experience (professional and academic) with software engineering including programming and databases. However, my teaching experience is limited to third-level education, not second-level. Before this study, the last time I had been inside a

post-primary school, was the last day of my Leaving Certificate. I was also unfamiliar with teaching Computational Thinking without using a programming language.

4.3.2.2 Teacher Experience

My initial site visits to post-primary schools and informal discussions with teachers in September/October 2018 highlighted a misinformed view of Computational Thinking, due to lack of understanding. This view was consistent with an exploratory survey that I issued through the CESI (Computers in Education Society of Ireland) mailing list on October 25th 2018. This survey had thirty-three participants. Of these respondents, sixteen were post-primary teachers, fourteen stating that they ‘agreed’ or ‘strongly agreed’ with the statement: ‘I have previous experience of a computer programming language’. Ten of these (post-primary) teachers (62%) stated that they would not be confident in teaching Computational Thinking. Their reasons fit into two categories: lack of knowledge about Computational Thinking and lack of knowledge about how to teach Computational Thinking. A sample of answers to the question: ‘Would you feel confident teaching a course on Computational Thinking?’ includes

Would need a support or training to get me started.

I love the idea and I have tried it but it is hard to get time and activities which I feel I need to teach it effectively.

A lack of understanding of exactly what is meant by computational thinking.

Don’t know what it is.

I know few problem solving sources and coding but not sure about the whole course...

4.3.2.3 Irish Policy Documents and Computational Thinking Content

A review of the Irish policy documents (see chapter 2 section 2.5) showed that the Irish Government’s approach towards Computational Thinking in compulsory education is very much linked to Computer Science, especially coding. The Junior Cycle Short Course Coding, and the Leaving Certificate Computer Science are subjects in the Irish second-level curriculum. Computational thinking is a topic in both courses (National Council for Curriculum and Assessment (NCCA), 2016; NCCA and DES, 2018). However, the rollout

of these subjects is not without problems (see chapter 1 section 1.2). There is also limited uptake in schools (see chapter 1 section 1.2). Computational Thinking is currently not a topic that many Irish post-primary teachers formally teach. It can also be suggested that there is a lack of information surrounding Irish students' subject knowledge on Computational Thinking.

4.3.2.4 School Setup

Real-world educational settings involve 'multiple dependent variables' (Barab and Squire, 2004, p. 3). Thus it was envisaged and subsequently verified (see Chapter 6) that many differences and variables would be encountered in the participating schools, including timetable availability, students' and teachers' knowledge on the subject, students' gender and ages, and school resources. A methodology that catered to these differences was needed. An approach that would focus on the characterising of variables rather than isolating them was required. EDR was judged to be that methodology. Kelly (2006) succinctly describes the relationship between design research and variables which highlights EDR specific approach to research:

Design research does not strive for "context free" claims; rather, it sees context as central to its conceptual terrain. ...It thus does not seek to "randomize away" these influences (classifying them as "nuisance variables"), but to engage, understand, and influence them in an act of co-design with teachers and students around the learning of significant subject matter. Thus, design research is not concerned with isolating variables... (Kelly, 2006, pp. 113–114)

4.3.3 Design Research Limitations

Design research is not without flaws. In her seminal article on design experiments, Ann Brown (1992) outlines a number of concerns with her new approach. The first is the overwhelming amounts of data collected; this can result in data not being analysed due to time constraints. Related to this concern is the 'Bartlett effect', that is, researcher selection bias where data is selected that supports his/her theoretical stance (Brown, 1992, p. 162). The Hawthorne effect is also highlighted as a concern. This is defined as an intervention having positive results due to subjects changing their behaviour as a result of their

participation in the research (Brown, 1992). Kelly (2006) adds to these concerns, stating that inadequate attention is paid to response and sampling bias in design research. He also critiques the lack of methodological rigour (there are no set methods), clear standards and argumentative grammar associated with this process (Kelly, 2004). Argumentative grammar is defined as ‘the logic that guides the use of a method and that supports reasoning about its data’ (Kelly, 2004, p. 118). McKenny *et al.* (2006) refer to the following dilemmas: 1) the researcher playing multiple roles and 2) the problems associated with real-world settings, that is, you lose control over data collection rigour, as the researcher interests are not the only ones at stake.

The above concerns have been addressed in this study as follows. To guard academic rigour, context-rich discussions are provided of the settings, design decisions and research results (McKenney, Nieveen and Van den Akker, 2006). As a researcher, I am aware of my multiple roles in this study: designer, researcher, teacher and observer. Putnam and Borko (2000) highlight the importance of acknowledging these roles:

Rather than pretending to be objective observers, we must be careful to consider our role in influencing and shaping the phenomena we study. This issue is obvious when individuals take on multiple roles of researchers, teachers, and teachers of teachers (2000, p. 13).

To reduce my conflict of interest, I acknowledged my impact on the study, triangulated collected data and implemented outside critiques of the intervention (McKenney, Nieveen and Van den Akker, 2006). Brown (1992) responds to criticism on design experiments in relation to the Hawthorne effect. She asserts that there is a significant difference between the original work by Hawthorne (and what is credited to it). Her observation refers to several points, of note she argues that in the original study, improvements only occurred when 1) workers understood that there were changes, 2) they perceived these changes were in their interest and 3) that they understood they were in control of said changes. She argues that in education interventions, this learner empowerment is what you strive for.

McKenny *et al.*, (2006) approach this topic differently, positing that the Hawthorne effect may be lessened by ensuring the research setting is as natural and genuine as possible. All my research (excluding online surveys, and desk work) is conducted in Irish schools.

In response to the argument on sampling bias, I developed a recruitment policy. This policy addressed the sampling bias concern and also the following concerns from academic literature (Nieveen and Folmer, 2013), for example:

- The clarification of participants' role, i.e., are they fulfilling the part of learner, critic or revisor, or are they playing more than one role?
- The selected participants' ability to help answer the research questions.
- Selective sampling. This is recommended for design-based research as it ensures that feedback is as information-rich as possible (Nieveen and Folmer, 2013)

Finally, in response to Kelly's (2004) critique of the lack of argumentative grammar associated with design research, a conjecture map was developed (Sandoval, 2014).

4.3.4 Mapping: Conjecture Map

Conjecture mapping was first proposed by Sandoval as a 'means of specifying theoretically salient features of a learning environment design and mapping out how they are predicted to work together to produce desired outcomes' (2014, p. 19).

Figure 4-3 displays the conjecture map for this study. It captures the design trajectory of this study and its most salient design features (Van der Linden, Van der Meij and McKenney, 2019). This study has two outcomes: an intervention (a Computational Thinking course), and a local instructional theory set. The intervention was designed with the following principles in mind:

- Flexible Design. The course should work for 40, 60 or 80-minute classes.
- Easy to follow. The course should be usable for both teachers and students.

- Low threshold. This concerns resources and pre-requisite knowledge. The course should be able to be taught from a standard classroom in a post-primary school. It is envisaged that teachers will need minimal pre-requisite Computer Science and programming knowledge to teach this PhD study's course (see 5.3).

The course should have the following characteristics:

- Effective (see 5.1.3)
- Engaging (see 5.2)
- Practical (see 5.1.2)
- Provides a foundation to the Computational Thinking component of the Leaving Certificate course (see 5.6)
- A high-quality course (see 5.1)

It was conjectured that these characteristics would be observed if certain design elements were followed. They are:

- Unplugged activities (see 5.3.1)
- Merrill's First Principles (see 5.4.1)
- Group work (see 3.3.2.1)

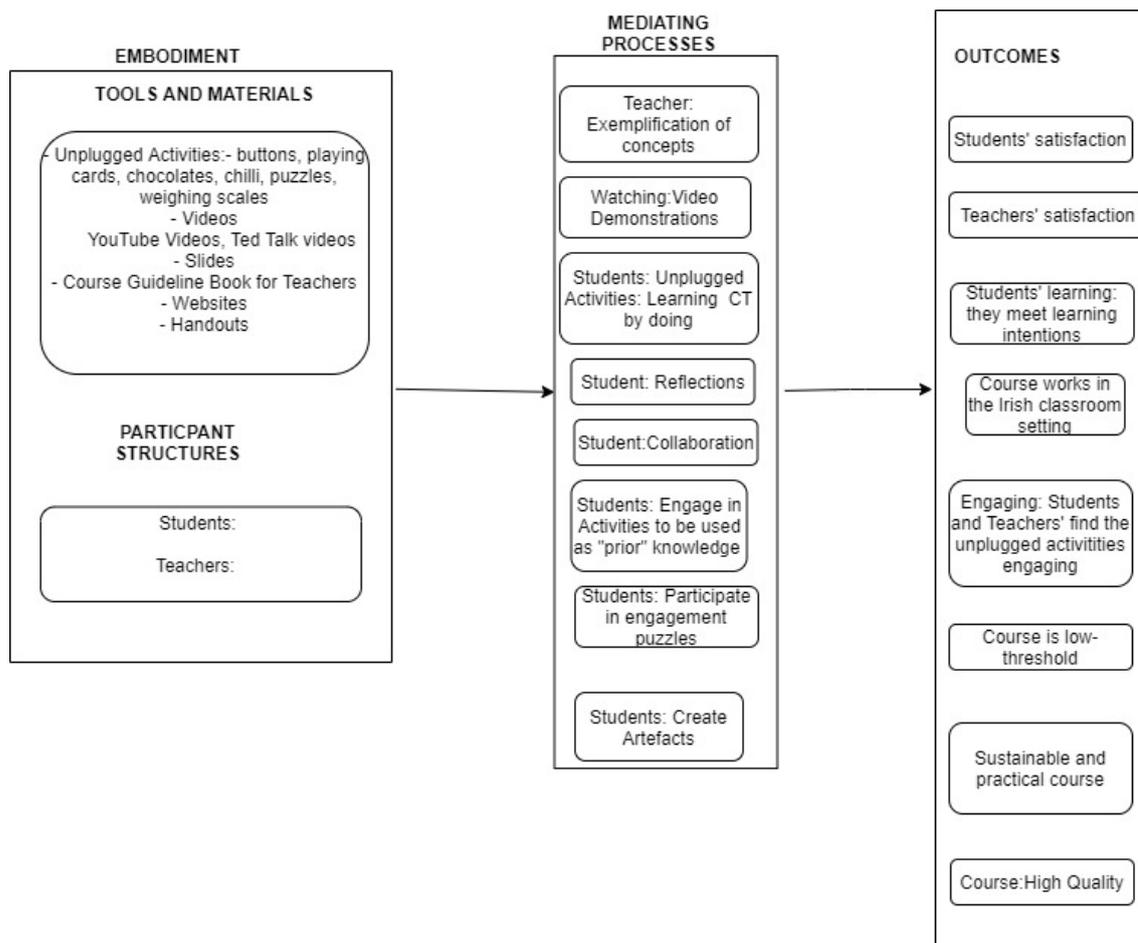


Figure 4-3 Conjecture Map for the Study

The high-level conjectures outlined in Figure 4-3 become reified within an embodiment. In this instance, the conjectures are embodied within two elements: tools and materials, and participant structures. The tools refer to the materials described in Chapter 5 (unplugged activities, videos, online games); the participants are Irish second-level teachers and students. These aforementioned embodiments generate specific mediating processes in the participants such as reflection, collaboration, engagement with puzzles and unplugged activities. These mediating processes should result in the desired outcome, in this instance, a high quality, engaging, effective course. The researcher's understanding of how embodied elements produce mediating processes is articulated as design conjectures. Similarly, the researcher's knowledge of how mediating processes result in the required outcomes is expressed as theoretical conjectures (Sandoval, 2014).

4.4 Design Research Process

The following section describes the design research process. The aforementioned conjecture map (see Figure 4-3) was modified to capture activities related to the EDR process, with the goal of providing structure to this process. As a researcher, I needed to see the intersection of the intervention's design trajectory with the EDR process, specifically the changing roles and actions of participants (including myself). The map also served as an articulation of my research work.

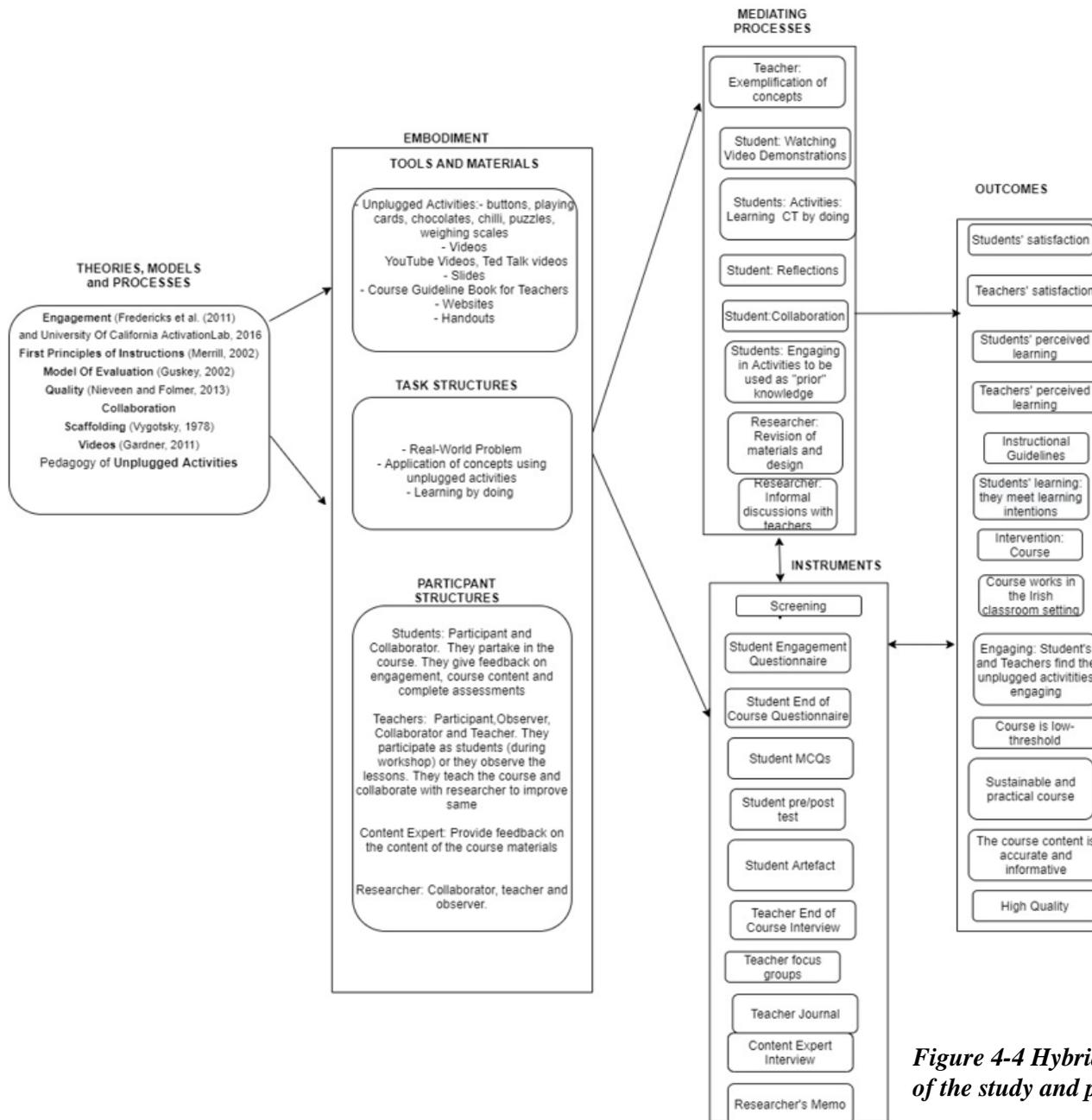


Figure 4-4 Hybrid Map that illustrates the design trajectory of the study and pertinent EDR activities

The conjecture map (see Figure 4-3) was modified as follows. The ‘Theories, Models and Processes’ box was added to highlight its influence on the created tools and materials (see embodiment box). Whilst the intervention and framework was revised during the EDR process; it was initially conceived and developed with theoretical input. The contents of the ‘participant structures’ box were modified to outline the roles played by the research participants. The instrument box was added to capture the research tools used to evaluate if the mediating processes produced the desired outcomes. This map serves to outline the intersections between the design trajectory of the intervention and the design of the research approach.

The following section describes the design research type and the approaches undertaken in this study.

4.4.1 Design Research: Type

Design research studies are conceptually categorised as being either development or evaluative (Nieveen, McKenney and Van den Akker, 2006). Development studies are defined as

the systematic analysis, design and evaluation of educational interventions with the dual aim of generating research-based solutions for complex problems in educational practice, and advancing our knowledge about the characteristics of these interventions and the processes of designing and developing them. (Plomp, 2013, p. 16)

Whilst validation studies are defined as the study of ‘educational interventions (such as learning processes, learning environments and the like) with the purpose to develop or validate theories about such processes and how these can be designed’ (Plomp, 2013, p. 15).

However, Plomp (2013) acknowledges that development and evaluation studies are often combined in practice. This is true of this study. While it is essentially a development study, the aim is to develop an engaging, practical, effective Computational Thinking course.

This study also aims to validate whether unplugged activities can be successfully used to teach Computational Thinking. The outputs of this study, and their descriptions are summarised in Table 4-1.

Table 4-1 Outputs from this study

Output	Type	Description
Intervention	Development	Design an intervention with the following characteristics <ul style="list-style-type: none"> • Engaging (Fredricks, Blumenfeld and Paris, 2004) • High quality (Nieveen, 1999b; Plomp, 2013) • Practical (Nieveen, 1999b; Plomp, 2013) • Effective (Guskey, 2002) • Low threshold
Design principles	Development	Local instructional theory. This will provide insight into the: <ul style="list-style-type: none"> • guidelines on the design process • purpose and characteristics of the Computational Thinking curriculum/course • empirical and theoretical arguments (proof) for the features and procedural guidelines. (Nieveen and Folmer, 2013)
Unplugged Activities	Validation	How are teachers successfully embedding unplugged activities to teach computational thinking? (Waite, 2017, p. 48)

4.4.2 Design Research: Phases

Commentators differ in how they envisage a design research project, but consensus exists that it contains several phases. Plomp (2013) distinguishes the following phases: preliminary investigations or front-end analysis, development or prototype development, and summative evaluation.

The preliminary research phase consists of the following activities, context and needs analysis; literature reviews; and conceptual or theoretical frameworks. The prototype phase is an iterative design phase. It consists of micro-cycles of research (see section 4.4.4.2) coupled with formative evaluations, with the goal being to improve and refine the intervention. The summative phase is the ‘final’ assessment phase, where you conclude whether the intervention has met its needs. This phase can also be considered the semi-summative phase, as it often results in recommendations for further improvements to the intervention and design (Plomp, 2013).

These three phases very much influenced my approach to this study. Figure 4-5 illustrates the activities that occurred in each phase. Each of these phases is discussed in the next section (4.4.3).

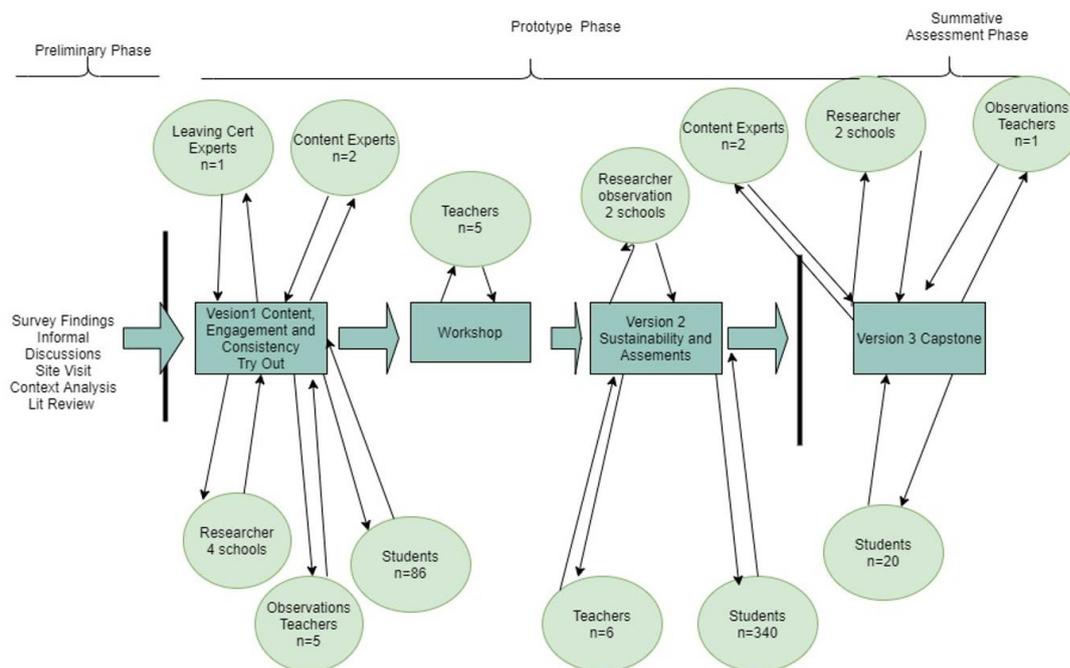


Figure 4-5 Phases and Activities that occurred during the EDR study (diagram format influenced by Mafumiko (2006) and Masole (2011))

4.4.3 Preliminary Analysis



Figure 4-6 Illustration of the three phases of EDR

The purpose of this section is to summarise the preliminary analysis phase of this study. Whilst the core elements of this phase and their findings have individually been discussed in detail in previous chapters, the goal here is to describe how the outputs from these activities combined to form the initial content and design for the Computational Thinking course. The activities that formed this phase were the literature review, media analysis, baseline survey and site visits. The purpose of these activities was to 1) establish the problem: the current context of Computational Thinking in the educational sector and 2) identify the initial design principles: local instructional theory and the initial theoretical framework for this study (Merrill's principles and unplugged activities). These outputs combined to provide input to the first prototype of the Computational Thinking course (Version 1).

This was achieved as follows: the literature review provided information on Computational Thinking's current knowledge base. It especially helped identify the existing interventions, pedagogies, and materials that could serve as sources of inspiration and provide lessons learned for my curriculum content (Nieveen and Folmer, 2013). This was especially relevant concerning unplugged activities. It helped filter out the insights from educational research that could be used in my study's design.

The media and policy document analysis provided insight into the current context of Computational Thinking in the Irish educational sector, and how Computational Thinking is perceived in the public consciousness, specifically it being perceived to be synonymous with coding. Site visits and conversations with post-primary school teachers provided

insight into my intervention's scope, e.g. the teachers' needs and abilities, the schools' facilities, and teachers' motivations (Nieveen and Folmer, 2013).

4.4.4 Prototype phase

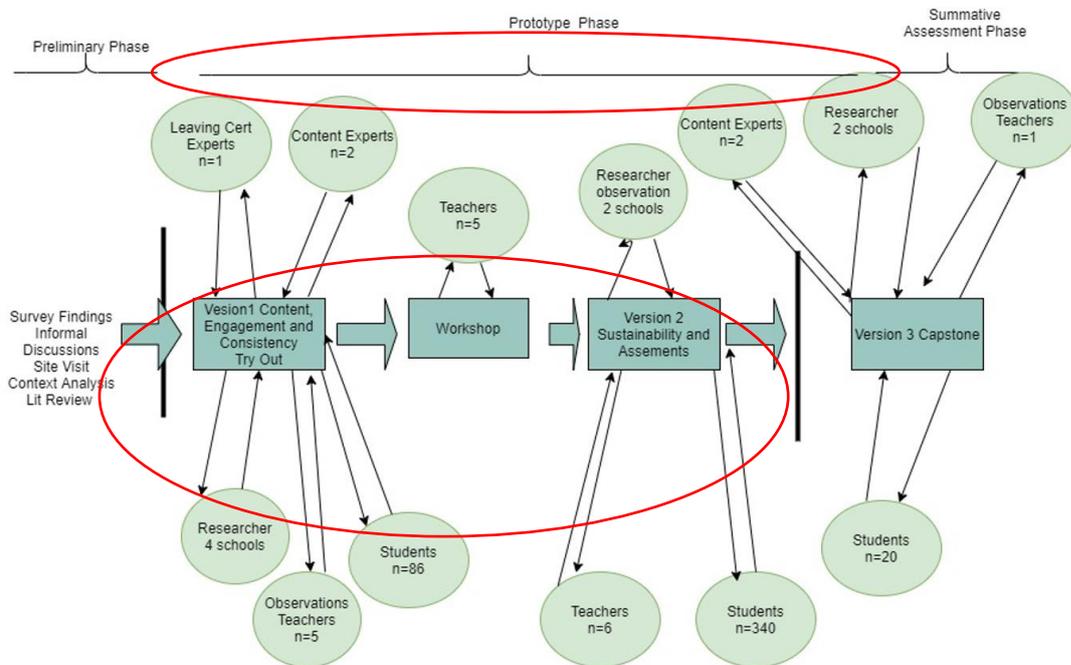


Figure 4-7 Prototype Phase of the EDR study

This phase consisted of the prototyping of the Computational Thinking course. The course was initially designed to be five weeks in length and delivered over 80 minutes per week. It contained five topics. The course was developed in full before it was piloted, as timetabling constraints within the participating schools and schools' expectations did not facilitate an approach where different lessons, pedagogies and activities could be trialed on an ad-hoc basis. Two versions of the course were developed and piloted during this phase, although micro-cycles occurred within each version. The reason for this is the timeline allowed for the content and delivery methods of lessons to be reviewed and modified between the piloting of a version in different participating schools.

The design model for this phase, whilst uniquely designed to meet my research environment's requirements, was influenced by Nieveen and Folmer's (2013) (see Figure

4-8). Their model illustrates how the prototype phase contains iterative cycles of analysis, evaluation and reflection.

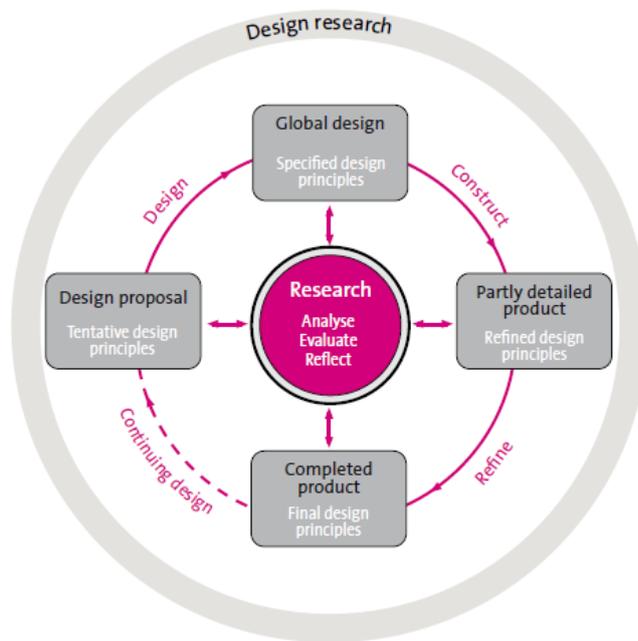


Figure 4-8 Development of Prototype phase (Nieveen and Folmer, 2013) (copyright CC by attribution)

This PhD study's prototype phase started with two components: Version 1 of the course and a set of tentative design principles. Version 1 was piloted in four schools with cycles of analysis, design and formative evaluation conducted after each school pilot. This resulted in the development of Version 2 of the course, and the refining of the initial instructional guidelines. A summary of the activities performed in Version 1 and Version 2 are described below (see section 4.4.4.1, and 4.4.4.4).

4.4.4.1 Version One

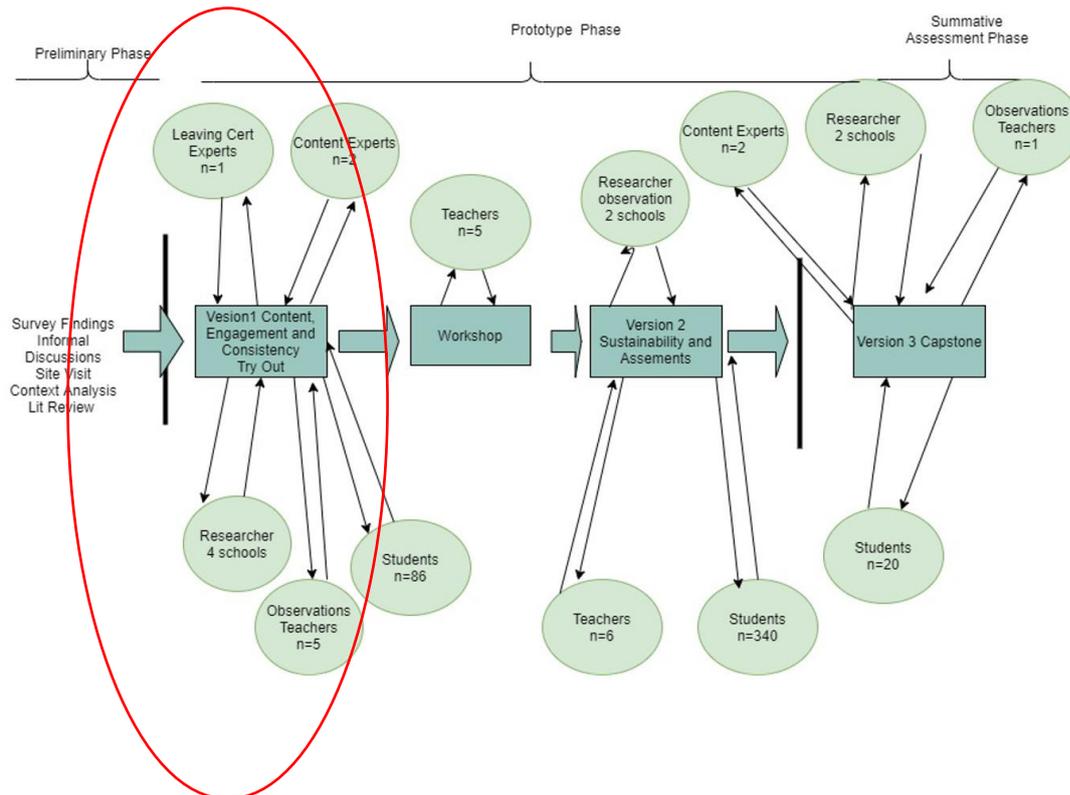


Figure 4-9 Overview of Version One

Version 1 was piloted in four schools: two all-boy schools, one all-girl and one mixed school. The timeline for this activity is displayed in Table 4-2.

Table 4-2 Version 1 timeline

School Type	Dates: 2019	Lesson length
School One: All boy	25 th Feb, 4 th March, 11 th March, 25 th March, 8 th April	80 minutes
School Two: Mixed	28 th Feb, 7 th March, 28 th March, 11 th April, 9 th May	40 minutes
School Three: All girl	8 th March, 15 th March, 22 March, 29 th March, 5 th April, 12 th April	80 minutes
School Four: All boy	12 th March, 19 th March, 26 th March, 2 April,	120 minutes

The goals for this Version 1 were to ascertain the following

- Engagement: Are the lessons, i.e., content and unplugged activities, engaging to students?
- Practicality: Is the course suitable for a school setting? Is the content usable and appropriate for the teachers?
- Effectiveness: Are the teachers and students satisfied with the course? What were their reactions? What did the students learn?

Version 1 of the course was taught by me, the researcher, with the participatory teacher(s) acting as an observer. Full details of this process are discussed in Chapter 6. In summary, the course consisted of five units. All five units were taught in three schools, with one school being taught four topics on a scaled-down version.

4.4.4.2 Micro-Cycles

The piloting of the course in the different schools (see Figure 4-10) occurred in parallel; thus, micro-cycles occurred within each version.

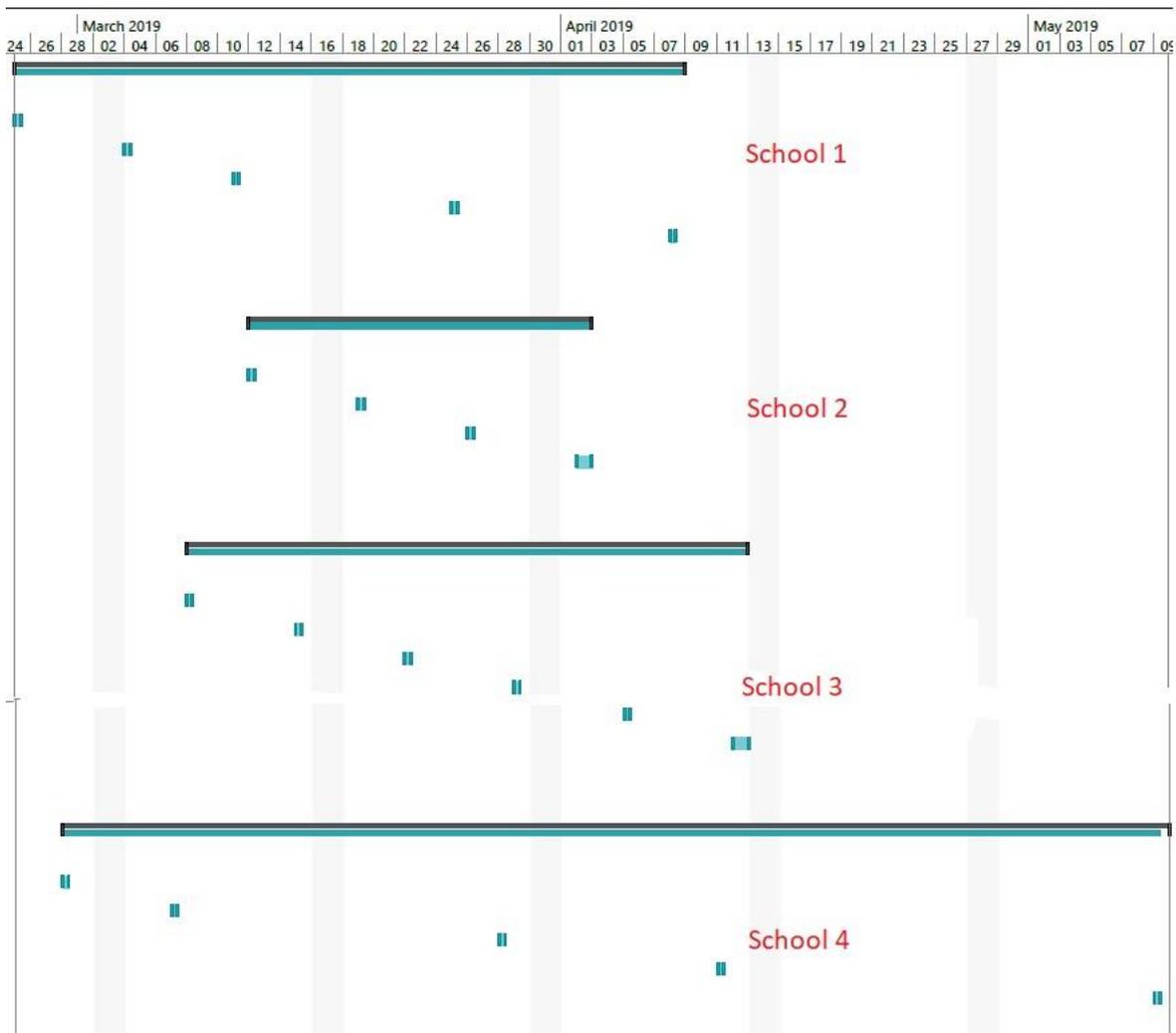


Figure 4-10 Gantt Chart showing the timeline of the lessons

Kennedy-Clark (2013) defines a micro-cycle in design research as a stand-alone study that focuses on fine-tuning a specific aspect of the research. For this EDR study, the term micro-cycle was applied to a lesson. Figure 4-10 shows a Gantt Chart that illustrates the microcycles, i.e. the lessons in each school. Each lesson is represented as a bar on the chart, with the continuous bar representing the full (macro) cycle (which is the course). The timeline allowed for the content and delivery methods of lessons to be reviewed and modified between piloting a ‘micro-cycle’/lesson in different participating schools. If a lesson or part of a lesson was unsuccessful in one school (see 6.5.1.1), it was changed when tried out in the next school. These micro-cycles informed the emerging local instructional theory (see Chapter 6).

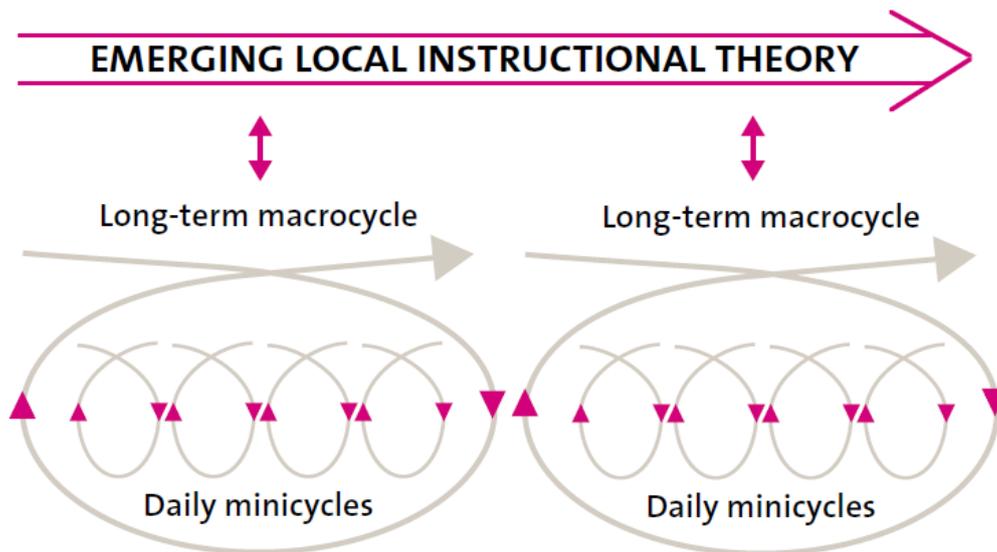


Figure 4-11 Diagram highlighting the microcycles in the development approach (Gravemeijer and Cobb, 2006) (copyright CC by attribution)

Formative evaluations occurred at the end of the course in each school. These are discussed in full in the data collection section (see section 4.7).

4.4.4.3 Content Experts

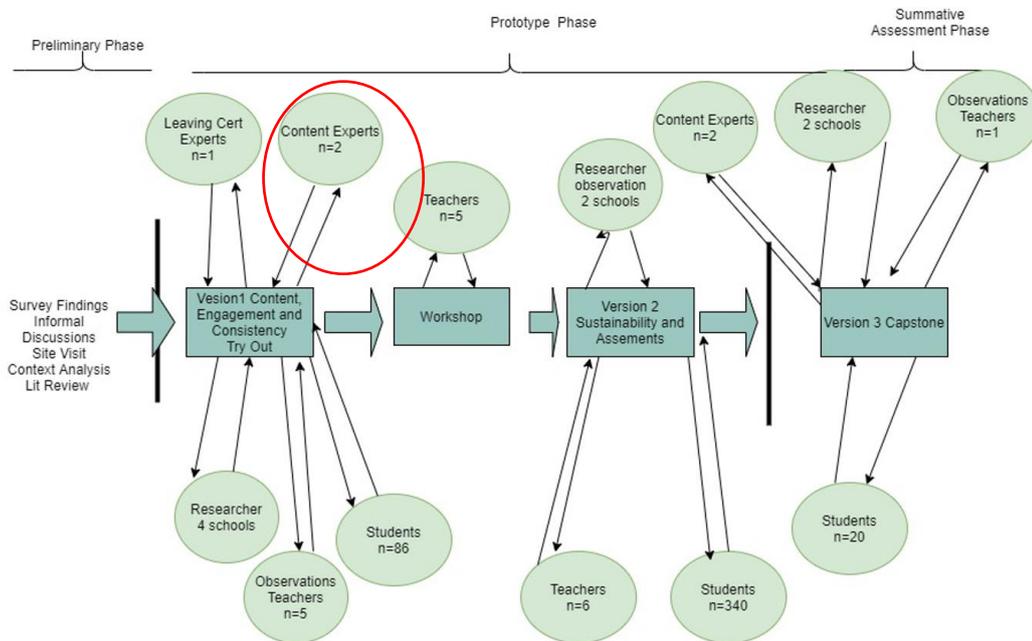


Figure 4-12 EDR process diagram. The involvement of the Content Experts in Version 1

A teacher’s guide was developed and reviewed by two content experts (see 6.5.1.3.1).

This process is explained in more detail in the quality section of this chapter (see 6.5.1.3).

It had been initially envisaged that the Computational Thinking experts' review would occur before the course was piloted in the schools, but timetable restraints did not facilitate this.

4.4.4.4 Version 2

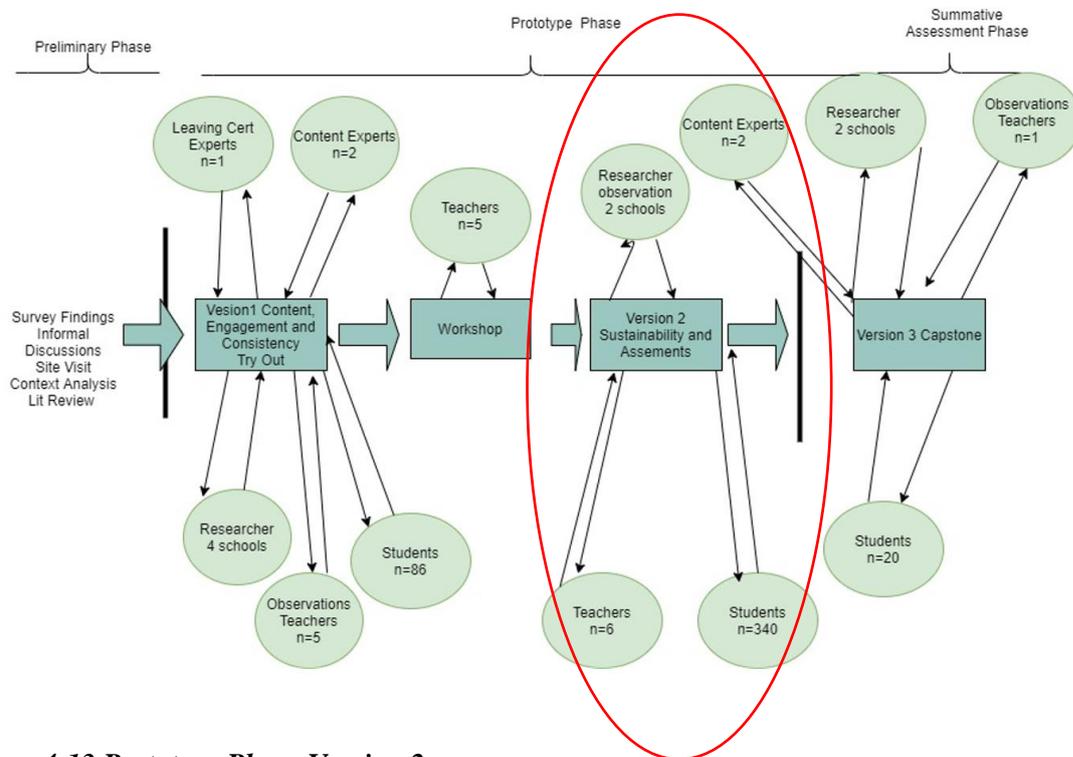


Figure 4-13 Prototype Phase Version 2

Version 2 was developed by refining Version 1 using both the collective formative evaluations results and feedback from 1) the students and teachers from the four participating schools, 2) the content experts, and 3) my own reflections. These changes are documented in Chapter 6.

The focus for Version 2 was the sustainability of the design in relation to how usable it is in practice and assessing learning. Two schools participated in this phase, both were all-boy schools, self-selected to participate in this research and were located outside Dublin. School details are provided in Table 4-3. Version 2 of the course was taught by the participating teachers, to assess the sustainability and effectiveness of the course’s design and content. Three hundred and forty students and six teachers took part in this phase.

Table 4-3 Version 2 timeline and details

School Type	Teachers Numbers	Classes	Student Numbers	Class Length
All-Boy	5	First Year, Second Year, Third Year, Transition Year	305	Two forty-minute classes. Run over eight weeks.
All-Boy	1	Transition Year	35	Two forty-minute classes. Run over eight weeks.

4.4.5 Semi Summative Phase

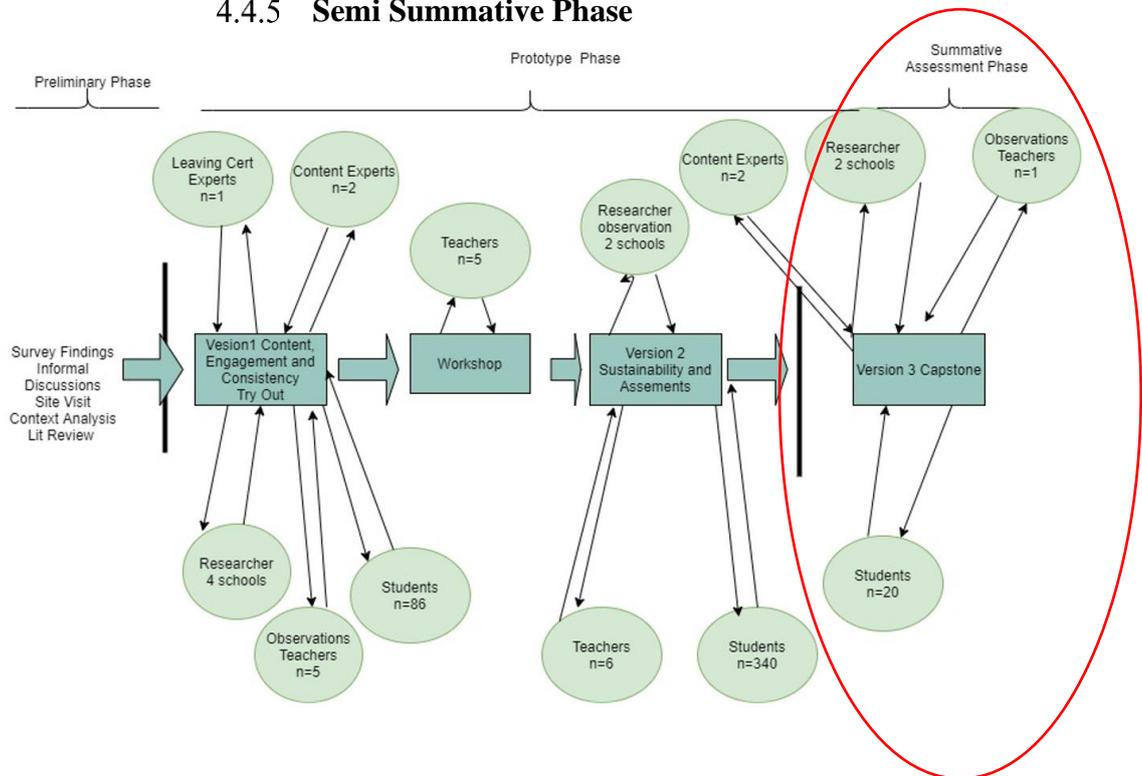


Figure 4-14 Version 3 of the intervention

This phase was conducted in two schools: an all-girls school and an all-boys school. Both schools were located in the Dublin area. I taught the course. This course served to implement changes identified by the teachers in Version 2. There was a time overlap between the completion of Version 2 and the start of Version 3 of the course.

Table 4-4 Timeline and details of the implementation of Version 3

School Type	Teachers Observer	Classes	Student Numbers	Class Dates (2019)	Class Length
All-Girl	1	Transition Year Fifth Year	15	Nov 6, 13, 20, 27 and Dec 4	80 minutes
All-Boy	1	Transition Year	5	Dec 2, 9, 10,17	3x 80 minutes with 1x120 minute class

4.4.5.1 Content Experts

Version 3 of the course was also reviewed by two content experts. Feedback from both experts is discussed in Chapter 8.

4.5 Data Collection: Evaluations

This section details the evaluations used in this study to assess the course's effectiveness and explore teachers and students' perceptions and feedback. Evaluations are 'the systematic assessment of the worth or merit of some object' (The Joint Committee on Standards for Educational Evaluation, 1994, p. 3). With reference to this study, formative evaluations occurred after Version 1, Version 2 and Version 3 of the prototype. The purpose of these systematic activities was to establish the 'worth' of the prototype. This was to evaluate if it was meeting its design goals, to ensure that an engaging, effective, practical course was being developed and to improve its quality (Nieveen, 2010). Unlike merit, worth is not context-free; it is 'grounded in field studies of local contexts and pluralistic value system' (Lincoln and Guba, 1980, p. 70). Thus, this PhD study's formative evaluations consider students and teachers' perceptions from each school combined with the content experts, and my perceptions. To assist in developing these formative evaluations, the evaluation matchbook developed by Nieveen, Folmer and Vligen (2012) was influential in establishing the evaluation framework and providing information on evaluation methods (see Figure 4-15). The methods used were interviews, try-outs, journaling, observations, screening, portfolios, and questionnaires.

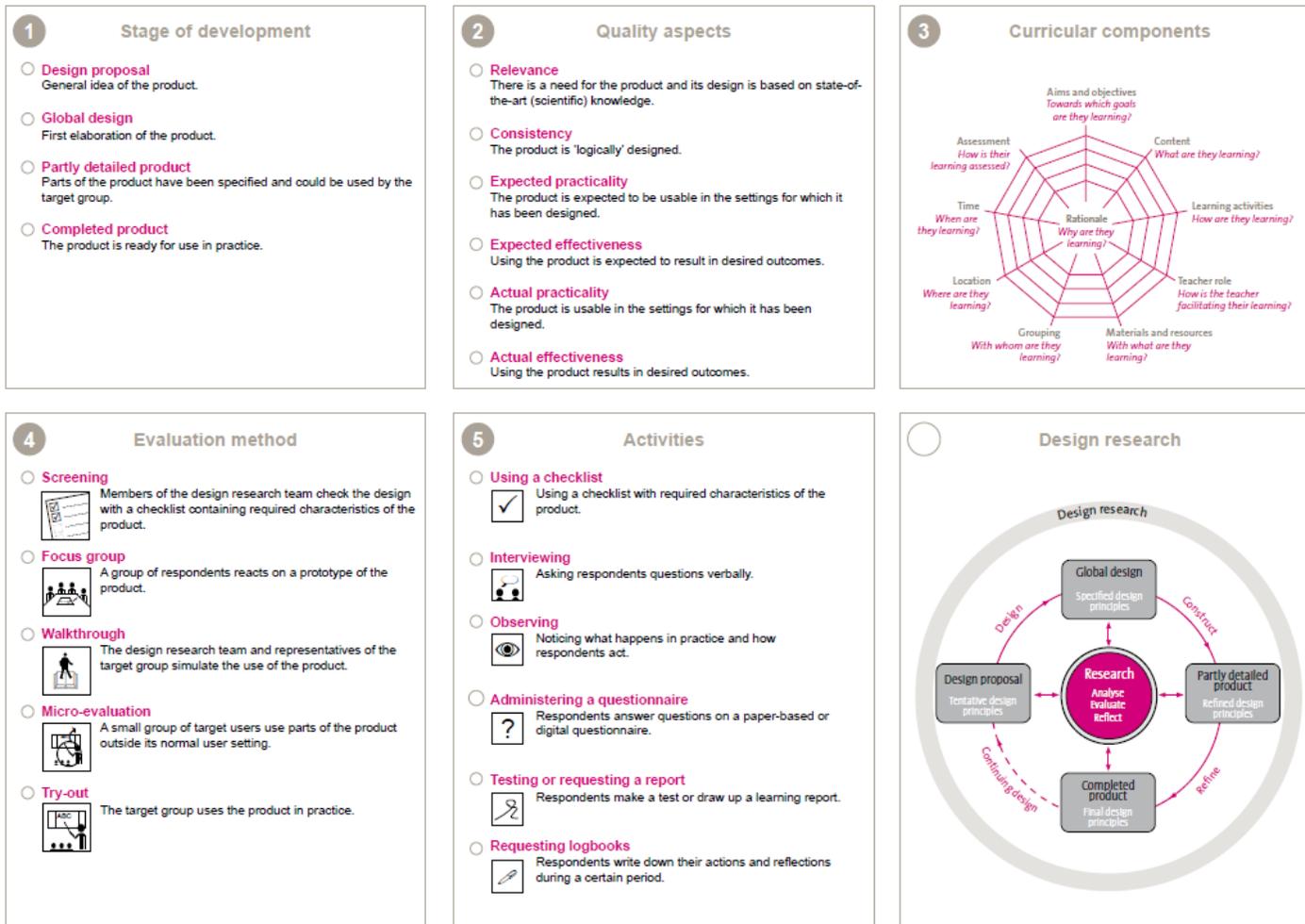


Figure 4-15 Evaluation matchbook developed by Nieveen, Folmer and Vligen (2012)

Table 4-5 A summary of the evaluation activities and corresponding instruments

Version	Criteria	Design Guidelines	Evaluation
1, 3	Engagement	<ol style="list-style-type: none"> 1. Curriculum designed based on Merrill's (2002) principles of instruction. 2. Used Unplugged Activities 	<ol style="list-style-type: none"> 1. Screening (researcher) 2. Students' questionnaire on Merrill's principles 3. Students' engagement questionnaire 4. Interview Teachers 5. Teacher's observations in journal 6. Researcher's observations
1, 2, 3	Effectiveness:	Curriculum designed based on Merrill's (2002) principles of Instruction. Guskey (2003) Participants Reactions Participants Learning Participants' Use of New Knowledge and Skills Student Learning Outcomes' (Goal is CT awareness and knowledge)	This was evaluated based on Guskey's (2003) model of effectiveness <ol style="list-style-type: none"> 1. Student questionnaires 2. Portfolios 3. Teacher interviews 4. Teacher's observations in journal
1, 2, 3	Low Threshold	Unplugged Activities	Interview (teachers) Student questionnaire feedback
1, 2, 3	Flexibility	Try out different schools with students of various expertise in this area	Researcher Observation
2, 3	Assessment	MCQ design Open questions on content	Pre and post Tests for Students Teacher interviews
2	Sustainability	Teachers teach course	Teacher Interviews and Teacher's observations in journal
1,2,3	Quality	Content Validity Content Consistency Practicality	See Table 4-11

		Effectiveness	
--	--	---------------	--

4.5.1 Screening

Screening is an evaluation method described as the process of checking the design with a checklist for required characteristics.

Screening was used to ensure the original course design adhered to Merrill’s First Principles of Instructions. Research has shown that following these principles results in effective, efficient and engaging learning (Merrill, 2002, 2009; Gardner, 2011; Gardner and Belland, 2012; Lo and Hew, 2017). To evaluate if the initial design captured Merrill’s (2002) First Principles, screening was used. A checklist was developed for each lesson of the course; it was greatly influenced by the questions on Gardner’s (2011) worksheet. The checklist used in this study to ensure lesson one captured all five principles (activation, application, integration, demonstration and real-world problem solving) is provided below.

Table 4-6 Checklist developed for Merrill’s First Principles of Instructions

Activities	Purpose
Two Logic Puzzles (warm-up puzzle)	To highlight that having a strategy is better than guesswork
Reflection (Integration)	Reflect on activity
Decomposition, Abstraction and pattern matching (Demo : video, activation : exemplification)	Teacher demo of decomposition abstraction and pattern matching in Real Life examples using videos, Lord of the Rings, London Underground. Examples given using real-world context
Spelling Test activity (Application using real-world problem)	Student activity of practising decomposition, abstraction and pattern matching together. This is a scaffolded exercise
Reflection (Integration)	Students reflect on activity
Spelling activity (Demo)	This is a demo to show how the Spelling Test solution works on a computer. This is a real-world problem

4.5.1.1 Screening and Assessment

A checklist was also developed to map each lesson's learning outcomes to the activities of the class and Bloom's Taxonomy (Bloom, 1956). A snippet of this checklist is displayed below

Lesson 1:

Table 4-7 Snippet of the Learning Outcome description table for Lesson 1

Learning Outcome Number	Learning Outcomes
L0T1-1	<ul style="list-style-type: none"> Critically assess problem-solving using both a strategy and guesswork
L0T1-2	<ul style="list-style-type: none"> Identify the components of the Computational Thinking framework
L0T1-3	<ul style="list-style-type: none"> Apply elements of the Computational Thinking framework to help solve a problem.

Table 4-8 Snippet of the assessments developed for Lesson 1

Number	Name	Description	Assessment Type
1A-1	Week 1 MCQ1	MCQ CT Description	GDrive Form Topic1_PreTest. MCQ
1A-2	Week 1 MCQ2	Decomposition Q	MCQ
1A-3	Week 1 MCQ3	CT Description	MCQ

Table 4-9 Snippet of the table that maps assessments and learning outcomes with Bloom's Taxonomy

Activity	Learning Outcome	Bloom (B) /Skill/Affective	Description
1A-1	L0T1-2	B Comprehension	MCQ
1A-2	L0T1-2	B Comprehension	MCQ
1A-3	L0T1-2	B Application	MCQ
1A-4	L0T1-2	B Knowledge	MCQ

4.5.2 Questionnaires

The questionnaire is an information-gathering technique where a collection of questions can be administered to participants. It is defined as ‘a technique for gathering statistical information about the attributes, attitudes, or actions of a population by a structured set of questions’ (Preston, 2009, p. 46). It is judged appropriate as a research instrument when respondents possess the required information and have the ability to answer the questions. Questionnaires are relatively quick to complete and comparatively easy to analyse (Cohen, Manion and Morrison, 2011). They are inappropriate for collecting information on sensitive topics, as the structure and brevity of the social encounters (between researchers and respondents) do not encourage trust and intimacy. This study used three questionnaires: an engagement questionnaire, an end of course perception questionnaire, and a workshop evaluation questionnaire.

4.5.2.1 *Engagement Questionnaire*

This questionnaire was issued to students in the Version 1 prototype phase, and the semi-summative phase (see Appendix B). It captured students’ opinions on their engagement with each lesson (five lessons in total). The captured data was triangulated with my researcher notes and teachers’ observations to facilitate changes to the content before that same lesson was piloted in a different school. The questionnaire used was self-reporting and adapted from the University of California’s Activation Lab Engagement Survey (2016). It had eight items and a five-point Likert scale. Three items captured affective engagement, two captured cognitive engagement, and the remaining three captured behavioural engagement. The rationale for its use is as follows. It captured all three facets of engagement, was quick to use and eliminates memory bias. The tool was previously tested for readability (Dorph, Cannady and Schunn, 2016; University Of California ActivationLab, 2016) and captured students’ subjective data on their engagement. Interviewing students after each lesson would not have been feasible due to time

restrictions and timetabling constraints. The goal here was to quickly capture the students’ voice after each lesson to highlight if there were serious issues with the engagement of the content that would require changes before the next ‘microcycle/lesson’ in a different school. In short, it was used for descriptive purposes, which informed the ‘on the fly’ analysis (see 4.7.1) conducted while the course was being piloted.

Student Post Lesson Questionnaire

Please indicate your level of agreement with the following statements.

- | | | | | | |
|--|--------------------------|-----------------|----------------|--------------|-----------------------|
| 1. During this lesson: I felt bored. | <i>Strongly Disagree</i> | <i>Disagree</i> | <i>Neutral</i> | <i>Agree</i> | <i>Strongly Agree</i> |
| 2. During this lesson: I felt happy. | <i>Strongly Disagree</i> | <i>Disagree</i> | <i>Neutral</i> | <i>Agree</i> | <i>Strongly Agree</i> |
| 3. During this lesson: I felt excited. | <i>Strongly Disagree</i> | <i>Disagree</i> | <i>Neutral</i> | <i>Agree</i> | <i>Strongly Agree</i> |

Figure 4-16 Snapshot of the post-lesson engagement questionnaire

As stated earlier, the instrument was adapted from the University of California’s Activation lab Engagement Survey (2016). The adaptations were as follows: the word *activity* in each item was changed to *lesson*, and a neutral option was added to the Likert scale. Garland (1991) suggests that the midpoint inclusion in a study is very much due to a researcher’s individual preference, which is very much the case here. I did not want to force students to commit to a certain position, thus reducing response bias (Brown, 2000; Croasmun and Ostrom, 2011).

4.5.2.2 End of course Perception Questionnaire

This questionnaire was issued to students in the prototype phase for Version 1 (by paper) and in the semi-summative phase with Version 3 (electronically) (see Appendix C). It was given out on the last day of the course. The questionnaire items were adapted from

Frick *et al.*'s (2009) Teaching and Learning Quality instrument for higher education students. Their instrument contained questions related to: Kirkpatrick's model on effectiveness (Guskey's model is an adaption of Kirkpatrick's model (Guskey, 2016)), Merrill's principles of instruction, and the BEST form (global items of quality used in Indiana University).

My questionnaire had twenty-one items, a five-point Likert scale and three open-ended questions. It collected data on Merrill's five principles of instructions, unplugged activities, the course effectiveness, student learning and students' opinion regarding the strength/weaknesses of the course. For example, items related to the demonstration and application of learning (Merrill, 2002) are as follows.

Demonstration

Media used in this course (for e.g. videos, websites,slides) were helpful in learning.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

My teacher gave examples of concepts that I was expected to learn.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

Figure 4-17 Snippet of 'End of Course' Questionnaire showing demonstration items

Application

I had opportunities to practice what I learned in this course.

Strongly Disagree Disagree Neutral Agree Strongly Agree

In this course, I was able to get feedback on my work

Strongly Disagree Disagree Neutral Agree Strongly Agree

Figure 4-18 Snippet of the End of Course Questionnaire showing application items

Students' views on the unplugged activities were captured using the following items

Unplugged Evaluation

The activities used in this course were helpful in learning.

Strongly Disagree Disagree Neutral Agree Strongly Agree

The activities helped increase my knowledge and skills in Computational Thinking.

Strongly Disagree Disagree Neutral Agree Strongly Agree

Figure 4-19 Snippet of the End of Course Questionnaire showing items related to unplugged activities

The effectiveness of the course was captured using items related to Guskey's five data levels for effective evaluation (2002, 2003, 2016). This questionnaire captured data on two levels: student's reactions and student's learning outcomes. The items that captured this are as follows:

- Overall, I would rate the quality of this course as outstanding.
- Overall, I would recommend this course to other students.
- I am dissatisfied with the activities used in this course (reversed)
- This course was a waste of time. (reversed)
- I am very satisfied with the content of this course.
- Compared to what I knew before I took this course, I learned a lot.

- I did not learn a lot in this course. (reversed)

Students were also asked three open-ended questions. They are as follows: What do you think you learned during the course? What are the strengths of this course? Do you have any specific recommendations for improving this course? These questions also served to capture data on student's reactions and student's learning outcomes.

4.5.2.3 Workshop Questionnaire

A workshop was conducted before the prototyping of Version 2. Its primary purpose was to teach teachers, who had not previously participated in the prototype of Version 1 of the course. A questionnaire was issued at the end of the workshop (see Appendix N). Its purpose was to evaluate the workshop and also to capture data on the course's quality (see quality section), teachers' reactions to the course (effectiveness (Guskey, 2002)) and their confidence in teaching said course. It was issued to five teachers. It began with three demographic questions followed by nineteen Likert scale survey items. It concluded with one question followed by three open-ended statements. It contained statements on the course content and materials, course satisfaction and instructional presentation. The questionnaire's design was influenced by student satisfaction questionnaires issued at third level institutions (Harvey, 2003) in particular the questions available online from academia such as the University of Wisconsin-Madison (no date), and the University of Pittsburgh (2021) library for teaching and learning. Satisfaction questionnaires from industry were also consulted and some questions used (Gravic Inc, 2015).

Using a 5-point Likert scale, teachers rated how strongly they agreed with each statement ((1) Strongly Disagree; (2) Disagree; (3) Neutral; (4) Agree; (5) Strongly Agree). A Cronbach alpha reliability analysis was run on the group items, but SPSS returned an error because only five participants answered very similarly.

The open-ended questions were as follows:

Do you feel confident to teach this course on Computational Thinking? If No, please outline how this may be rectified.

Please elaborate on your answers to the following statements

- a. The course content is appropriate for first-year students.
- b. The course content meets its goal of assuming that no previous Computer Science experience is necessary for teachers.
- c. The course meets your needs.

4.5.3 Pre and Post Tests

Pre and post-tests were issued to students at the start and end of all course implementations.

4.5.3.1 Prototype Phase Version 1:

During the prototype phase of Version 1, the test issued consisted of six open-ended questions and three demographic questions (see Appendix D). Their purpose was to ascertain students' awareness of the following topics pre and post course: Computational Thinking, Artificial Intelligence and Algorithms. These topics were judged to be essential as 1) they play an important role in the course and 2) there are challenges connected with their teaching and definitions (Sentance and Csizmadia, 2017; Holmes, Bialik and Fadel, 2019) (see 6.6.1.4.3).

These tests served to inform the course's instructional design by highlighting misconceptions or instructional difficulties and by gathering data related to Guskey's (2002) students' learning outcomes data level. The tests also provided feedback on the course content by providing a baseline on students' knowledge of these subjects, as it was unknown.

The questions on Computational Thinking and algorithms are depicted in Table 4-10.

Table 4-10 Question 1 and Question 3 issued in the pre and post test for Version 1 of the course

Pre-Test	Post-Test
<p>Q1</p> <p>a) What do you think Computational Thinking might mean?</p> <p>b) Can you describe any situations/problems that might use Computational Thinking?</p>	<p>Q1</p> <p>a) What is Computational Thinking?</p> <p>b) Describe in simple terms how would you use some (or all) the components of Computational Thinking to help solve the following problem, -- Creating a study timetable</p>
<p>Q3</p> <p>a) Explain what you understand by algorithms?</p> <p>b) Give or guess some examples of algorithms.</p>	<p>Q3</p> <p>a) Explain what you understand by algorithms?</p> <p>b) Give or guess some examples of algorithms.</p>

4.5.3.2 Prototype Phase Version 2 and Semi Summative Phase:

Multiple choice questions (MCQ) were issued to students during the prototype phase Version 2 and the semi-summative phase, Version 3. The questions were devised to test the learning outcomes of each topic. The assessments formed part of an assessment strategy that was influenced by Bloom’s Taxonomy (see Chapter 6 and section 4.5.1.1).

In Version 2 of the course, the MCQs were issued pre and post-topic. This resulted in ten tests. Students answered forty questions in total (with twenty questions being unique).

The rationale for pre and post topic MCQs is that it would reduce memory bias. The tests had been planned to be given online, but paper tests were issued due to technical difficulties (see Chapter 7). In Version 3, students were given tests pre and post-course (as they had an engagement post-lesson questionnaire). This resulted in only two MCQ tests, which contained twenty-one unique questions. They were given online using google forms (see Chapter 8).

Select the best answer. Artificial intelligence is concerned with

- Making a computer intelligent by increasing its memory
- false intelligence
- machines that learn by feedback and pattern matching
- machines that are self reliant i.e. need no human intervention and are self aware

The term Ethics is concerned with

- Making decisions that are valid
- Making decisions that are moral i.e. good and bad
- Making decisions that generate money

Figure 4-20 Snapshot of the MCQ test given to students online.

4.5.4 Journals/Emails

The journals and emails took the form of research-driven diaries. A research-driven diary is a data collection method that can be used to collect quantitative and qualitative data.

When data is written in prose form, it is usually supplemented by a diary-interview. They can be considered as an alternative to data observations (Cohen, Manion and Morrison, 2011)

As observers, teachers were asked to record observations regarding student engagement and course content. As course teachers, they were asked to record weekly their experience teaching the course. Their entries were written in prose format. Of the eleven teachers who participated in the study, all six teachers who taught the course diligently sent weekly, informative emails. Regarding the teachers who observed the course only, three preferred to give feedback verbally, with only one teacher using the journal. This resulted in me recording their feedback in my researcher notes and confirming the same at the interview.

4.5.5 **Researcher Notes**

My researcher notes had the same function as the research-driven diary (see 4.5.4). My entries were written in prose format and ‘away from the situation’ (Cohen, Manion and Morrison, 2011, p. 235). After each school visit and microcycle, I wrote up my observations and reflections on same in a Word document. Depending on the situation, I would also record my observations (in the school car park) and transcribe them later.

4.5.6 **Teachers Interviews**

Interviews are the most widely used method in qualitative research. They allow for the collection of subjective information on attitudes, values, beliefs, information and events. They are a flexible tool (Cohen, Manion and Morrison, 2011). Individual and group interviews were used to capture teacher data in this study. The data captured provided insight into teachers’ perspectives when acting in the role of observer and teacher. They also served to ascertain teachers’ opinions on student engagement, student learning, course practicality, sustainability, course content, and they also validated my researcher notes.

Individual interviews were conducted at the end of all course implementations (i.e. Version 1, 2, 3). Group and individual interviews were also conducted during the Prototype Phase Version 2, with one school.

Eleven teacher interviews were conducted. The ‘end of course’ individual interviews were semi-structured. This allowed flexibility to explore issues in the teachers’ journals/emails, and provided a safeguard to ensure questions related to the research aims were pursued (Bryman, 2012). A sample of some of the semi-structured questions asked is provided below, these questions are related to Content, Teaching, Teacher Learning, Student Learning, Engagement and Experience. The questions were derived from questions posed by Guskey (2002) in acquiring data related to his five levels.

4.5.6.1 *Sample Interview Questions*

The following questions were asked to teachers after the piloting of Version 1 of the course:

- Is the content of the Computational Thinking curriculum age-appropriate for Transition Year students?
- Is the content of the curriculum practical for a Transition Year setting?
- Is the content engaging for Transition Year students?
- Does the content meet its goal of assuming that no previous Computer Science experience is necessary?

The following questions were asked to teachers after the piloting of Version 2 of the course:

Content:

- Did you like the content?
- What would you improve?
- Did you think it was appropriate for the students?

Teaching:

- How did you find teaching this course?
- What were the challenges?
- Would you teach it again?
- What would you change?

The group interviews occurred in a school staff room. At times they were non-directive with teachers sharing their experiences with each other, and I listened. Other times, they were directive as I posed questions based on their weekly emails. The group interviews

occurred over three school visits and consisted of two to five teachers at any one time. A sample of questions are

- How are you getting on?
- Tell me more about your classroom management issues?
- How are you finding the content?

4.5.7 Survey

A survey is a data collection method that ‘gathers data at a particular point in time with the intention of describing the nature of existing conditions, or identifying standards against which existing conditions can be compared, or determining the relationships that exist between specific events’ (Cohen, Manion and Morrison, 2011, p. 256). A survey was issued to members of the CESI mailing list to ascertain their understanding and attitude towards Computational Thinking. This survey also served as a recruitment tool for the implementation of the course. A survey instrument was used as it is a tried and tested way to understand and measure attitudes and behavioural responses (Chew and Eysenbach, 2010). The survey was adapted from Yadav *et al.*'s (2014) Computing Attitude Survey. Thirty-three individuals completed it. It consisted of six open-ended questions, three Yes/No questions; these questions were concerned with school resources and individuals' understanding of Computational Thinking. It also contained nineteen Likert items which were concerned with attitudes towards Computer Science and Computational Thinking. It concluded with five demographic questions.

An example of the questions are as follows

Q8 Please indicate your level of agreement with the following statements

	Strongly Disagree	Disagree	Agree	Strongly Agree
Knowledge of computing will allow me to secure a better job	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
My career goals do not require that I learn computing skills	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I expect that learning computing skills will help me to achieve my career goals	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
Having background knowledge and understanding of Computer Science is valuable in and of itself	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I do not think it is possible to apply computing knowledge to solve other problems	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I am not comfortable with learning computing concepts	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I can achieve good grades (C or better) in computing courses	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I can learn to understand computing concepts	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I doubt that I have the skills to solve problems by using computer applications	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
I think Computer Science is boring	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
The challenge of solving problems using Computer Science appeals to me	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Figure 4-21 A screenshot of the questionnaire

4.5.8 Student Artefacts

Artefacts are objects that ‘can be seen, heard, smelt, touched, felt, even tasted and heard’ (Cohen, Manion and Morrison, 2011, p. 532). Their use by researchers depends on the research question. During the implementation of the course, artefacts that were developed as part of the course's assessment policy or as part of completing an unplugged activity were collected, see Chapter 6.

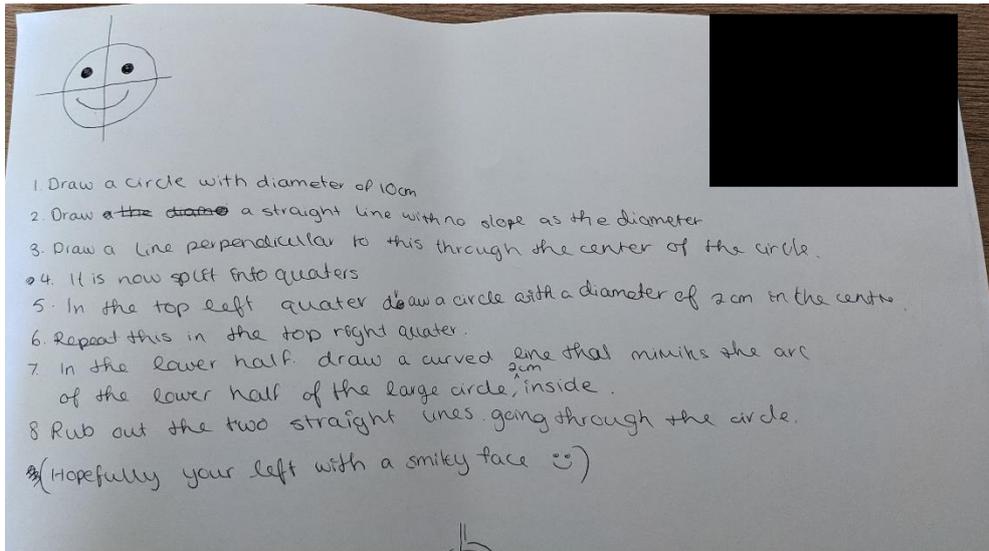


Figure 4-22 Student Artefact from Unit 2: Writing an algorithm to draw a smiley face emoji

4.6 Data Collection: Quality

Closely linked with the evaluation of a course is the quality of the course. Nieveen (1999a) proposed three generic measures associated with quality: validity, practicality and effectiveness. This is explained in detail in Chapter 5 (see 5.1). Table 4-11 summarises the instruments used to evaluate the quality of the course.

Table 4-11 Instruments used to gather data on course quality

Criteria Variable	Sub-Variable	Design Phase	Evaluation	Instrument
Quality	Relevancy Content	Version 1 Version 2 Version 3	Interview content experts *4 Interview teachers	Interview (teacher) Interview (content expert)
	Relevancy Consistency	Version 1 Version 2 Version 3	Interview experts Interview teachers Workshop questionnaire	Interview (teacher) Interview (content expert) questionnaire
	Practicality	Version 1 Version 2	Teacher journals Researcher journal Interview Teacher	Journals Interviews
	Effectiveness	Version 1 Version 2 Version 3	See below	
	Viability: Practicality, relevance and sustainability	Version 1 Version 2	Teachers can teach course from the teacher's guide Relevance: foundation to CS	Computer Science Leaving Certificate experts consulted

Effective Model	Participants reactions 'Did they like it? Was their time well spent? Did the material make sense? Will it be useful?' (Guskey, 2002)	Version 1 Version 2* Version 3 *2 focus is teachers	Student Questionnaire after course Teacher interview/teacher journal They think it was a worthwhile course	Questionnaire (students) Interviews, journals, workshop questionnaire (teachers)
	Participants Learning 'Did participants effectively apply the new knowledge and skills?' (Guskey, 2002)	Version 1 Version 2* Version 3 *V2 focus is teachers	Pre and post assessment Portfolio of students learning Teachers' learning	Student Assessment Interview (teachers) Workshop Questionnaire (teachers)
	Participants' Use of New Knowledge and Skills	Version 2	Teachers teaching the course Direct observation Student artefacts Teacher interviews	Interviews (teaching) Teacher journals
	Student Learning Outcomes	Version 1 Version 2* Version 3 *2 focus is teachers	Learning Outcomes Awareness Learning goals:	Pre and Post Tests (students) Teacher interviews End of course questionnaire (students)

4.6.1 Content Expert Interviews

Similar to the teachers, the data collected from the content experts was qualitative data, using semi-structured interviews. The guiding questions were influenced by questions from a design research study conducted by Fauzan, Plomp and Gravemeijer (2013). They are as following:

1. Can you confirm that you've read the course document and how fresh it is in your mind?
2. Can you allude to your knowledge on Computational Thinking?

Questions regarding the content of the course

3. Does the content of the curriculum include relevant subjects and topics? (By relevant I mean relevant to Irish Students from First to Transition year, and as a course designed to be five weeks in length, conducted 80 minutes a week over 5 weeks)
4. Does the content of the curriculum reflect the key principles of Computational Thinking?
5. Has the curriculum the potential for developing students' understanding of Computational Thinking?

4.7 Data Analysis

Data is gathered in this study using various means: individual and group interviews, teacher journals/emails, student artefacts, student multiple-choice pre and post-tests, student open-ended pre and post-tests, a survey and questionnaires (three types). The data analysis, similar to the data is mixed. It involves thematic and content analysis and descriptive statistics. The process used was non-standard in that I split the retrospective analysis stages into two parts. Whilst all analysis in design research is in essence, retrospective, a distinction is usually made between 'on the fly' analysis after each microcycle/lesson and retrospective analysis at the end of each full cycle (Bakker and van Eerde, 2015). The three-stage analysis conducted in this study was developed to uniquely meet this study's complexity, restrictions, and requirements. 'Because each qualitative study is unique, the analytical approach used will be unique' (Patton, 2015, p. 522). This resulted in three stages of analysis: 'on the fly', 'validation', and 'standard' analysis such as thematic and content analysis.

4.7.1 Stage 1 'On the fly' Analysis

'On the fly' analysis is the name given to the analysis that is performed after each microcycle. The purpose here was to change elements of the content and unplugged activities that did not work in that lesson before being piloted in another school. These adjustments form part of the EDR process. Bakker and Van Eerde (2015) highlight that

they have to be reported well as they are part of the data corpus. The data collection instruments were my field notes, informal discussions with teachers after lessons, teachers' emails/journals, and student engagement questionnaires. My researcher notes were verified at the teacher interviews.

4.7.2 Stage 2 Validation Analysis

Stage 2 Validation Analysis was introduced to this study, to cater for its unique needs. It is the name given to the analysis that occurred after the piloting of Version 1 of the course. The time between the finish of the pilot of Version 1 and the start of the pilot for Version 2 was tight. I had two months, July and August 2019, to conduct all my data analysis from the first implementation of the course, rewrite the course content and prepare for the teacher workshop. With that considerable task in mind, the qualitative data from the teachers, content experts and students were analysed in this stage to identify feedback that would inform, validate and improve the intervention. This was a top-down approach, in that it was research question driven (i.e. deductive) (Fereday and Muir-Cochrane, 2006). Three categories were predefined before the analysis commenced. They were Quality, Validation and Corrective Feedback. These categories were judged by me to best improve the intervention's validity, as they served to validate not only the content of 'Version 1' but also my design conjectures (Sandoval, 2014). This analysis resulted in actionable tasks. A description of the three categories used is described in Table 4-12.

Table 4-12 Categories used in the Validation Stage Analysis

Category	Description
Quality	<p>This data refers to</p> <ol style="list-style-type: none"> 1) the improvement of the instruments used (i.e. data given by the interviewee, which could improve data collection instruments such as a content expert recommending I ask students a specific question in a test) 2) data that confirms the interviewee has read the document 3) data that refers to the interviewee experience. <p>This data may result in actions.</p>

Validation	This category refers to data that validates the current approach, content, activities, ideas described in the course guide.
Corrective Feedback	The category refers to data that suggests actions regarding the content. This may take the form of correcting errors, simplifying activities, clarifying definitions, providing suggestions for reference or justification of the current stance.

The steps taken in this analysis phase were as follows.

1. The transcripts were professionally transcribed.
2. The transcripts were read several times in their entirety.
3. The text of each transcript was then subdivided into sections; each sentence in each section was closely examined for one of the following three categories: quality, validation, corrective feedback. Where appropriate, the text was highlighted, and a category assigned to it.
4. Three tables were next constructed for each category and the highlighted code was pasted into the appropriate table rows.

Quality Background	. Yes, I have taught computer studies at school, years and years ago but I haven't taught transition year in Ireland on a short course, so there's gaps in my thinking 
Quality Read document Tracked changes	I've read through this on the bus here from  for the record on your audio recording, I've got in front of me your Word document and I've made comments on it with tracked changes which I'm going to send to you

Figure 4-23 A snippet of the table devised from Expert 1 and Expert 2. This table contains the data categorised under the headings of Quality with the created subcode.

5. The data in the tables were re-analysed to create subcodes. For example, under the quality category, subcodes such as ‘read document’ ‘tracked changes’ were created. These subcodes allowed for similar content to be grouped together.
6. The second pass of the data in the tables resulted in creating a list of actionable tasks. These tasks (where appropriate) were actioned to improve the design and content of the course see Table 4-13.

Table 4-13 List of Actions taken to improve the Content of Version 1 of the course

Category: Sub Category:	Corrective Feedback List	Action Made
Quality: Interview Questions	Consider the word relevant in your interview question, relevant to what? Does the content of the curriculum include relevant subjects and topics?	I changed my interview script to accommodate this feedback. It now reads. Does the content of the curriculum include relevant subjects and topics? (By relevant, I mean relevant to Irish Students from First to Transition year, and as a course designed to be five weeks in length, conducted 80 minutes a week over five weeks)
Quality: Interview	Ensure I cover my review of content from multiple angles	I had four experts and eleven teachers read my course
Corrective Feedback Content Language:	Change the course document's title to give context, such as course length and the type of students it is aimed at.	I changed the title from ‘An introductory course to Computational Thinking’ to ‘A short introductory course in Computational Thinking for Transition Year students and Teachers’.

Quantitative data captured during the implementation of Version 1 using questionnaires was analysed as standard. This is discussed in section 4.7.4.

4.7.3 Stage 3 Thematic Analysis

Stage three of the analysis was the name given to the analysis that occurred after all versions of the course were piloted. It was conducted between April and October 2020. The purpose here was to revisit the qualitative data from all phases (prototype and

summative) to gain a deeper understanding of the students and teachers' perspectives on the course, specifically about engagement and effectiveness. Revisiting data bolsters the original analysis.

Stage 1 and Stage 2 analyses were action-based and focused on the intervention. The Stage 3 analysis provided a more in-depth insight into the data gathered, thus identifying the characteristics that contributed to or deterred in making the course engaging, effective, and low-threshold. This served to inform the local instructional theory. The teachers' email/journal notes and interviews as well as the content experts interviews, were re-analysed using thematic analysis. Thematic Analysis is defined as 'a method for systematically identifying, organizing and offering insight into patterns of meaning (themes) across a data set' (Braun and Clarke, 2012, p. 57). This allowed me to focus on a collective meaning of 'quality' across the data set. My strategy was to follow Braun and Clarke's (2006) six-step thematic analysis framework, using the software tool NVivo. The steps of Braun and Clarke's Thematic Analysis are as follows 1) become acquainted with the data, 2) generate initial codes 3) search for themes, 4) review themes 5) define and name themes, 6) write up analysis (Braun and Clarke, 2006). This process is defined in detail in section 6.5.1.3.3.

4.7.3.1 Coding Schemes in Stage 3

Different coding schemes were used to analyse the qualitative data in this stage. Saldana (2016) identifies seven broad first-cycle coding methods. I used two, grammatical and elemental.

4.7.3.1.1 Qualitative Data from prototype phase Version 1

The grammatical methods (i.e., sub-coding and simultaneous coding) and the elemental methods (structural and in Vivo coding) were used to capture the interview and journal/email data generated in prototype phase Version 1. The coding process was both deductive and inductive in that I started by creating the three original categories straight away, i.e. Corrective Feedback, Quality and Validations. I next worked inductively, generating initial codes.

4.7.3.1.2 Qualitative Data from prototype phase Version 2

The elemental method (descriptive coding) was used to analyse the teachers' emails, in the prototype phase Version 2 (Saldana, 2016). The emails/journals were very factual thus, they facilitated nouns or a short phrase to summarise their content. However, the teacher interviews were coded using both descriptive and in vivo coding. The in vivo method captured the emotions and richness exhibited during the interviews, thus providing a deeper understanding of the participants' minds (Saldana, 2016).

4.7.4 Questionnaires

4.7.4.1 *Engagement Questionnaire*

The collected data on lesson engagement was analysed as follows. The questionnaire was issued to students on paper, so I first re-created it on the Qualtrics software and manually entered each lesson's questionnaire results. All results were checked twice for errors.

Using the Qualtrics software, the data was exported to SPSS format and uploaded to said SPSS. I next changed the items' label names from the actual item statements (which is default in the import) to the following: E1_Bored*, E2_Happy, E3_Excited, C1_Daydreaming*, C2_Focused_Learning, B1_Busy_doing_other_tasks*, B2_Time_quickly, B3_Talked_to_others_not_related*. I reverse coded four of the

variable values, Q1, Q4, Q7 and Q8. These are the values marked with an asterisk. This process is shown in a snippet from the Qualtrics output file.

Here is a cut and paste from my output file.

```
DATASET ACTIVATE DataSet8.
RECODE Q1 Q4 Q7 Q8 (1=5) (2=4) (3=3) (4=2) (5=1) INTO Q1r Q4r Q7r Q8r.
EXECUTE.
```

Descriptive (frequency) statistics were then run on the collected data to identify any severe issues with the course content (related to engagement). The frequency statistics were mode and median values. The median value is the middle value of a list. The modal value is the most frequently occurring. They were used to depict central tendency, i.e. the average and most common response to the items by students. The median value is recommended for Likert scales, as the mean of ‘Agreed’ and ‘Neutral’ makes no sense, and the mean of extreme responses can be deceptive (Sullivan, Artino and Jr, 2013). The modal value is considered valuable for ordinal data, as it is unaffected by outliers (Cohen, Manion and Morrison, 2011). The mean and modal results were triangulated with data obtained from my field notes and the participating teachers' observations. They served to validate the content (see 5.2). Figure 4-24 depicts the median and mode values for school 1 in week 3 of the course.

Week 3 16 students	Bored[®]	Happy	Excited	Daydream[®]	Focused	Time Quick	Busy Other[®]	Talking[®]
Median	4.5	4	4	4	4	4	4	4
Mode	5	4	4	4	4	4	4	4

Figure 4-24 A screenshot of the frequency statistics captured for school 1 in week 3.

4.7.4.2 End of Course Questionnaire

Quantitative data from the End of Course questionnaire was inputted into the SPSS software, as the questionnaires were also completed by hand. The data for each student was checked twice for errors. Three items were reversed coded, and descriptive statistics were run on the collected data.

The questionnaire also contained three open-ended questions. These students' answers were scanned into the NVivo 12 software package. Their answers were short two to four lines and were read many times before they were coded using NVivo 12 software.

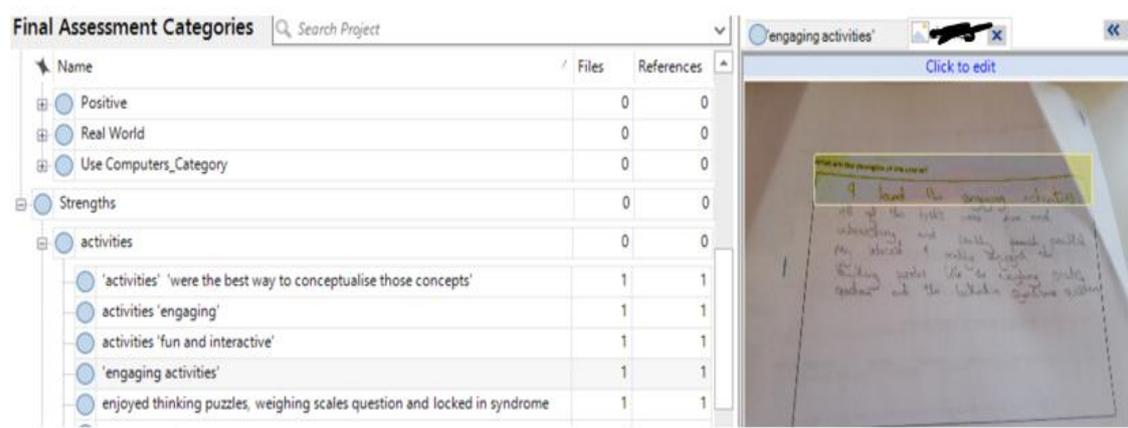


Figure 4-25 A screenshot showing a scanned questionnaire and the created categories in NVivo.

The analysis of this data was influenced by content analysis (i.e. conceptual content analysis), which is similar to thematic analysis. However, my focus was not to look for emerging themes but to conduct analysis that would be both descriptive and quantifiable (Vaismoradi, Turunen and Bondas, 2013; Sabharwal, Levine and Dagostino, 2016). The reason for this was due to personal experience in my own professional teaching practice. From programme board meetings, I am aware of how important it is to not rush into

making a change based on one person's recommendation when that change can result in you unknowingly upsetting the rest of the class. I thus wanted to quantify student answers. I also wanted to compare the results from different schools. I am aware that due to the nature of design research, (i.e. testing and revising conjectures and activities while the study is in progress), direct comparison between classes/schools is not possible. However, the answers to the open-ended questions were compared to see if an issue reported (by many) in one school was also reported in another school.

In the context of design research, it will not be sufficient to come to understand one student's thinking. Instead to be of value, the researcher must document that a significant proportion of students reason in a comparable manner. (Gravemeijer and Cobb, 2006, pp. 27–28)

4.7.4.2.1 Conceptual Content Analysis Steps

The steps taken were as follows.

1. Three categories were initially created in NVivo: Learning, Strengths and Recommendations. As before, this was a top-down approach (deductive), in that it was driven by the three open-ended questions (Fereday and Muir-Cochrane, 2006).
2. The students' answers were coded under each category. The coding strategy used was in vivo coding, as it results in rich, evocative codes (Saldana, 2016)

Name	Files	References
'shorten the classes'	1	1
'sparked an interest in computers and logical thinking'	1	1
Use Computers	4	4
Strengths	0	0
'activities' 'were the best way to conceptualise those concepts'	1	1
activities 'engaging'	1	1
activities 'fun and interactive'	1	1
'Algorithms .. help with certain situations through life'	1	1
collaboration	2	2
'different unique'	3	3
'engaging activities'	1	1
enjoyed thinking puzzles, weighing scales question and locked in syndro	1	1
'helps you think fast using logic'	1	1
'helps you think outside the box more and think about it logically'	1	1
'involved in everyday life although not many people know about it'	1	1
'liked' 'less like a class and more like a lecture' 'chilled out'	1	1
'loved' 'AI lesson'	1	1

Figure 4-26 A screenshot illustrating the *in vivo* coding scheme adopted for the open-ended questions

- Next, the codes were sub-categorised under each main category. For example, the sub-categories for Strengths were as follows: 'activities', 'class setup' etc. (see Figure 4-27).

Class Timing	0	0
Coding	0	0
Content	0	0
Positive	0	0
Real World	0	0
Use Computers_Category	0	0
Strengths	0	0
activities	0	0
Class setup	0	0
Different	0	0
Learning Strengths	0	0
'Algorithms .. help with certain situations th	1	1
'helps you think fast using logic'	1	1
'helps you think outside the box more and	1	1
'involved in everyday life although not man	1	1
'loved' 'AI lesson'	1	1
'thinking logically' 'it shows how to make c	1	1

Figure 4-27 The subcategories for the main category of Strengths.

- Review the subcodes. After further reflection, the ‘different’ and ‘class setup’ subcategories were merged into a setup category.

Setup	0	0
'teacher was easy going'	1	1
'Not something usually offered in sch	1	1
'liked' 'less like a class and more like a	1	1
collaboration	2	2
'different unique'	3	3

Figure 4-28 The Setup sub-category.

- The data was next quantified and displayed in a tabular format. Table 4-14 displays the descriptions, examples and frequencies of the sub-categories of Strengths.

Table 4-14 Descriptions, Examples and Frequencies of the Strengths sub-category for a participating school.

Content	Description	Example	Frequency Column	
			Mentions	By Person
Activities	Answers mentioned the activities, and their opinion of them	‘fun and interactive’ ‘I loved the engaging activities’ ‘peaked my interest’	9	5
Set Up	Answers relate to the structure of the class and also how the	‘I liked how it was less like a class and more like a lecture, it	7	6

	course was different and unique	was a bit more chilled out 'different' 'unique' 'teacher easy going' 'working with others was great'		
Learning	Answers relates to learning a particular subject and its contents	'Helps you think outside the box more and think about it more logically' 'It is something involved in everybody's life although not many people knew much about it'	6	6

4.7.5 Pre and Post Tests

In Prototype Phase Version 1, pre and post tests that consisted of six open-ended questions were issued to students. The students' answers for Q1, Q2 and Q3 were analysed and displayed using the steps described in the Lamprou and Repenning (2018) study, which was based on content analysis. The responses for each question were grouped according to several categories. These categories were created using the students' own answers. Student's answers were also graded using a rubric design influenced by Rodriguez *et al.*'s (2017) assessment of unplugged activities. His rubric had three categories: unsatisfactory, partially proficient and proficient. The students' responses were categorised as belonging to one of these categories based on the definition and content used in the course. Figure 4-29 shows the categories of students answers pre-course to the question 'What do you think Computational Thinking might mean?' Table 4-15 captures the same in tabular format accompanied with grade information and examples of responses. Figure 4-30 shows a sample of the rubric used to grade the students' answers.

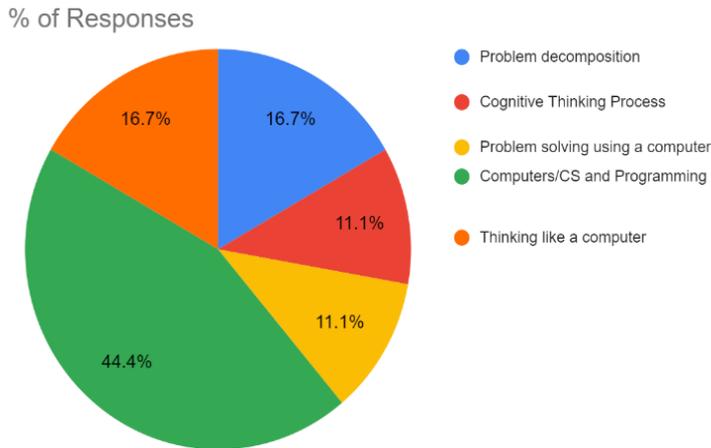


Figure 4-29 Categorised student answers to the question ‘What do you think Computational Thinking might mean?’

Table 4-15 Results and grades from the analysis of the open-ended question ‘What do you think Computational Thinking might mean?’

Category and Description	Example of Responses	%	Grade
Problem decomposition Answers identify CT as a process that involves breaking down a task into smaller steps	‘the process of breaking down a large task or a large goal into smaller tasks or steps which can easily be completed’ ‘Things are broken down’	3/18 16.7%	PP, PP
Cognitive Thinking Process Answers mention a specific thinking /cognitive process	‘applying Logic to any task’ ‘also might be logical thinking’	2/18 11.1%	PP, PP
Problem solving using a computer Answers identifying CT as a problem solving technique that is used in combination with computers. Thinking with the help of a computer.	‘Using the aid of computers to help solve day to day problems, Basically solving problems with the aid of technology’ e.g. ‘planning to build a city’	2/18 11.1%	U,U
Computers/CS and Programming Answers strongly connect CT with programming skills, Computer Science or how a computer works	‘Being able to program/write code’ ‘Thinking in coding’ ‘Thinking about how a computer works and what they can do’ ‘learning how a computer works’	8/18 44.4%	U,U, U,U,
Thinking like a computer Answers imply that a computer is a thinking body	‘thinking like a computer’ ‘computer is making certain decisions’	3/18 16.7%	U,U, U

	‘thinking like a computer to get a better understanding of it’		
--	--	--	--

CT Unsatisfactory	CT Partial proficient	CT Proficient
The student’s answer of CT did not subscribe in any part to what was taught in the course.	Student’s answer is partially correct in that their description of CT matches partly what was taught in that course but did not fully subscribe to the definition.	<p>Student’s answer subscribes to what was taught in the course. It recognises that CT is a thought process. That there are components of CT: decomposition, pattern matching, abstraction and algorithms. Logical thinking is also a part of CT as it helps to ensure the validity of our reasoning.</p> <p>Answers may also allude to the following <i>In Computer Science, when analysing and solving problems, one has to think a certain way. This reasoning is very much influenced by the way computers can be programmed</i></p>

Figure 4-30 Rubric for grading the question ‘What do you think Computational Thinking might mean?’

4.7.5.1 Multiple Choice Questions (MCQ)

Pre and post-multiple-choice questions were issued to students during the implementation of Version 2 and 3 of the course. Due to time constraints and difficulties encountered by teachers, only the MCQ from Version 3 of the study were analysed (see Appendix E).

4.8 Ethical Consideration

The ethical principle of ‘do no harm to participants’ underpinned my ethical approach to this study. This was achieved by adhering to the following principles documented in Bryman (2012): avoid harm, ensure informed consent, protect privacy and no deception.

4.8.1 Avoid harm to participants

I was very conscious when devising the Computational Thinking course to ensure both the unplugged activities or content did not cause students undue stress or affect their self-esteem. In particular, the content of ‘Unit 4: Ethics’ was discussed before delivery with participating teachers to ensure that it was age-appropriate and suitable.

I was also aware of ‘power relationships’ common in teacher-led research with students (Department of Children and Youth Affairs (DCYA), 2012). Coupled with this concern were issues relating to being an insider researcher. Being conscious that collaboration is a cornerstone of EDR, I continuously monitored myself to maintain ethical practices and commitments (Fleming, 2018).

Ethical permission was obtained from DCU’s Research Ethics Committee (reference number DCUREC/2018/201). I also completed the required ethical training course in DCU and underwent the Garda vetting procedure set out by DCU registry.

4.8.2 Ensure informed consent

Written consent was obtained from all participants: teachers, content experts, students and their guardians. This written consent was accompanied by a plain language statement that stated the aims and goals of this study.

4.8.3 Privacy

My participants were informed on who has access to their data: myself and my supervisors Dr Costello and Dr Donlon. Participants data was collected, stored and processed in compliance with General Data Protection Regulation. Before submitting ethical approval, the DCU Data Controller was sent a Personal Data Security Schedule (PDSS).

Participants' information was treated with confidentiality unless otherwise indicated (content experts permitted their names to be used). Where relevant, participants were assigned a pseudo-name, and any identifying information such as a person's name, place and school were removed. Data was stored securely at DCU, and also online using Dropbox, which automatically encrypts files.

4.8.4 Deception

There was no deception involved in this study.

4.9 Summary

This chapter provides a justification for the EDR methodology employed in this pragmatic study. It outlined how this methodology, coupled with specific research methods, served to address this PhD study's overarching research question. It summarised the processes undertaken as a part of EDR and described the data generated and analysed. It also stated the research ethics associated with this study. Chapter 5 is concerned with the initial design of the Computational Thinking course and subsequent framework.

5 Computational Thinking Course and Instructional Design

As mentioned previously, this EDR study has two outcomes: an intervention, the Computational Thinking course, and a set of guidelines, the local instruction theory for both the learning and teaching of Computational Thinking to Irish post-primary students. This chapter is concerned with discussing the Computational Thinking course: its aims, pedagogical approach, initial content and design, and rationale for the former. It is also concerned with the emerging local instructional theory that resulted from the course creation.

This research aims to answer the following question: what are the characteristics of a high quality, practical, engaging, effective, and low threshold course for both the learning and teaching of Computational Thinking to Irish second-level teachers and students? This PhD study also aims to validate whether unplugged activities can be successfully used to teach Computational Thinking.

This chapter starts with discussing what is understood by a course being practical, engaging, effective, high quality, and low threshold. It next discusses what is understood by unplugged activities. Finally, it discusses the initial instructional design model for this PhD's CT course, and the course contents.

5.1 Quality

An aim of the Computational Thinking course is that it is of high quality. In respect to EDR interventions, quality is defined as having three criteria, namely validity, practicality and effectiveness (Nieveen, 1999b)

5.1.1 Validity

An intervention is considered valid if it is relevant. It must be based on the state of the art knowledge (content validity). It must also be logically designed, with all its elements

linked with each other (construct validity) (Nieveen and Folmer, 2013). The validity of the Computational Thinking course was ascertained from the literature reviews (see Chapter 2 and 3) and it was also evaluated by content experts and teachers (see 6.5.1.3, 6.6.1.2). Section 5.6 discusses the course content and how it had its beginnings in Irish policy and educational curriculum documents.

5.1.2 **Practicality**

Practicality is defined as the intervention being ‘usable in the setting for which it has been designed and developed’ (Plomp, 2013, p. 29). The content needs to be usable and appropriate for the teachers. The unplugged activities need to work in a school setting.

5.1.3 **Effectiveness**

A course is considered effective, if it results in the desired outcomes (Plomp, 2013). Effectiveness is understood based on Guskey’s (2002) model, where he specifies the five levels of data to gather to evaluate a course’s effectiveness. This PhD study is concerned with four of those levels: Teacher Reactions, Teacher Learning, Teachers use of Knowledge and Skills and Student Learning Outcomes. The fifth level of data ‘Organisational Support and Change’ is not captured. Data concerned with student reactions and learning is also captured.

5.2 **Engagement**

Engagement is a characteristic that was important to this PhD from the onset. The course content and instructional design will be designed with this attribute in mind, and the course will be evaluated to see if this objective had been achieved. The reasons for this are manifold. Hidi and Renniger (2006) underscore how engagement is important in developing interest in a subject. Their studies highlight how positive experiences are important for the development of personal interest in a subject and growth in students’ belief in their proficiency. It is also shown to be an important variable in respect to

learning, although some studies highlight that this causal relationship is hard to prove (Axelson and Flick, 2011). Fredricks *et al.* (2011) state that engagement supports students' learning specifically concerning achievement, persistence in school and positive academic results. Carini *et al.*'s (2006) study corroborates the linkage between student engagement and desirable learning outcomes, in their case grades and critical thinking, but are careful to emphasize that student engagement is just one of the sources that influences learning outcomes. Axelson and Flick (2011) highlight that a causal relationship between engagement and learning is hard to prove, positing that it is difficult to control for all the variables. Nevertheless, they argue that it is better that students are engaged than not, and that some student subgroups benefit from engagement more than others, even if this cannot be correlated with increased learning. They also emphasize that students become disengaged when tasks become either too complex or too easy. Thus by designing the course with engagement in mind, it is envisaged that this will play a role in providing students with a positive experience of Computational Thinking. Similarly, by capturing students' and teachers' engagement with the lessons, it would help evaluate the content's appropriateness, especially regarding the pedagogy of unplugged activities.

5.2.1 How is engagement defined?

Similar to Computational Thinking, there is much variation in how engagement is defined and measured. Fredricks *et al.* (2004; 2011) proposed that student engagement has three dimensions: behavioural, emotional, and cognitive. Behavioural engagement is concerned with student participation and contribution to academic and extracurricular activities. Emotional engagement focuses on students' affective reactions for example: happiness, boredom, and interest. Cognitive engagement is defined as the level of exertion or effort students' invest in their learning. Dorph *et al.* (2016, p. 56) define engagement as 'one's focus, participation, and persistence within a task'. Similar to Fredricks *et al.* (2004;

2011), they conceptualised engagement as having the same three dimensions, but in relation to a task rather than a course. They see behavioural engagement as the behaviour related to completing a task; cognitive engagement as the student's thought processes or attention during the task; with affective engagement being the students' emotions during the completion of the task. Both Fredericks *et al.* (2011) and Dorph *et al.* (2016) definitions influenced how engagement is defined in this PhD study. For this PhD study, engagement is considered a multidimensional facet, containing the same three cognitive, behavioural and emotional elements, with students' engagement within a lesson and with the course (as a whole) being captured.

5.3 Low threshold

This concerns resources and pre-requisite knowledge. The course should be able to be taught from a standard classroom in a post-primary school. It is envisaged that teachers will need minimal pre-requisite Computer Science and programming knowledge to teach this course.

5.3.1 Unplugged Activities

Not all Irish students want to become computing professionals. Not all Irish students want to learn to code. However, they all live in a digital world. The course needs to have an approach that caters to students interested in pursuing Computer Science as a career and those who do not.

An approach to teaching Computational Thinking that does not rely on programming skills is advantageous for many reasons. It will increase the course's applicability and allow students to tackle complex problems in Computer Science (which relate to real-life) without pre-requisite knowledge. It will remove the difficulties surrounding programming, especially those found in novice programmers. For example, around

explicit looping and one-sided control flow (the else is never used) (Pane, Ratanamahatana and Myers, 2001; Guzdial, 2008). The focus will be on algorithmic solutions that humans can read and understand rather than machines (Curzon *et al.*, 2019). Coupled with the above reasons, are the resources needed to teach a programming language, which are discussed in the following section.

5.3.1.1 *Resources*

As part of the context analysis, school visits and informal conversations were conducted with post-primary teachers. This highlighted the difficulties associated with technology from the lack of WiFi and internet access for students (not teachers). Some schools were limited to one computer lab (or limited Chromebooks), which they stated were in great demand due to the new Junior Cycle curriculum and classroom-based assessments. They also spoke about different operating systems been found on the computers. The problems that were relayed mirrored many of the challenges found by Sentance and Csizmadia (2017) in their study on ‘Computing in the curriculum: Challenges and strategies from a teacher’s perspective’. Their findings highlighted the technical difficulties encountered by British teachers, such as software installation and technicians’ flexibility. The resources needed for this PhD course will be of low cost or readily available in the classroom.

5.3.1.2 *Teachers’ views of Unplugged Activities*

The five most commonly mentioned challenges encountered by British teachers when teaching computing is as follows:

Teachers’ own subject knowledge,
student’s lack of understanding of content,
technical problems,
differentiation to meet different level of abilities
students willingness or ability to solve problems (Sentance and Csizmadia, 2017, p. 479).

On the flip side, British teachers’ five most successful strategies when teaching computing are: unplugged activities, collaboration, computational thinking,

contextualisation of learning, and scaffolding programming. Unplugged activities were the most common and most used successful strategy mentioned. They are activities used to teach Computer Science away from a computer. In this study, they will also be used to teach Computational Thinking. This approach has merit as it is acceptable to teachers. In fact, with reference to teacher professional development, the following articles also highlighted the popularity of unplugged activities and Computational Thinking with teachers. Curzon *et al.* (2014) used unplugged activities embedded in storytelling to introduce Computational Thinking ideas to teachers and report that feedback was overwhelmingly positive with teachers. This mirrors Morreale *et al.*'s (2012) workshops aimed at teaching Computational Thinking to teachers; they included unplugged activities amongst their resources, with teachers reporting that being able to use resources immediately was a great advantage. However, how the activities transfer to the classroom is unknown.

5.3.1.3 *Gap in the Literature*

A literature review commissioned by the Royal Society (UK) on the subject of pedagogies in Computing Education had two clear intentions: to identify 'what is known about pedagogies for teaching computing in schools' and to identify 'potential gaps where useful pedagogy research could be carried out to support teaching computing in the UK' (Waite, 2017, p. 4). The following gaps in knowledge were highlighted concerning unplugged activities and Computational Thinking:

- how teachers are successfully embedding unplugged activities to teach computational thinking
- how teachers are successfully embedding unplugged activities to teach computer science concepts; (Waite, 2017, p. 48).

This mirrors Bell and Vahrenhold (2018) study that drew attention to how surprisingly few empirical studies have been conducted about how Computer Science unplugged

activities have been used in a regular classroom. However, much of the work on unplugged approaches is anecdotal and much more research is needed (Curzon 2019).

5.3.1.4 *Students' perspective on Unplugged Activities*

Students' views concerning unplugged activities are mixed. Taub, Armoni and Ben-Ari (2012) study on the 'Views, Attitudes, and Intentions' of middle school children towards Computer Science reported mixed views on the effectiveness of unplugged activities in this context. Students' understanding of Computer Science was partially improved, but their desire to study Computer Science was lessened. Taub *et al.* (2012) speculate that the unplugged activities may need to be more explicitly linked to computer concepts, as they currently stand they do not link to student's prior knowledge. Bell and Lodi concur with these speculations, stating:

unplugged activities are effective when used in a context where they will be ultimately linked to implementation on a digital device, either through programming, or by helping students to see where these ideas impinge on their daily life. (2019, p. 16)

Feaster *et al.* (2011) study using Computer Science unplugged programs on high school students showed that their outreach programme had no impact on student attitudes towards Computer Science and their perceived content understanding. They conjectured the following reasons 1) that high school students do not find kinesthetic activities exciting, 2) the students were already involved in Computer Science programming classes, and thus believed they were already experienced programmers. This study will address a gap in the literature regarding the use of unplugged activities in developing and evaluating teaching and learning pedagogies for second-level students. It will also contribute to knowledge through its outcomes by developing a course to fix a problem in the Irish second-level system regarding the teaching of Computational Thinking.

5.4 How will the Computational Course be designed?

Findings from the systematic literature review highlighted the prevalence and importance of collaboration and scaffolding (irrespective of the tool) when teaching Computational Thinking. Findings also identified task and concept based learning as an important technique when learning CT especially using unplugged activities (such as Bebras). The importance of linking the unplugged activities to computing was also highlighted to minimise student dissatisfaction. For reasons already discussed, the course will be designed to have the following characteristics: engagement, effectiveness, practicality, high quality, and low threshold. This research has two outcomes: an intervention (CT course) and a set of guidelines (local instruction theory). The course will need to be completed in full before piloting as timetabling constraints and schools' expectations do not facilitate a 'think big, start small' approach.

For this reason, a design model was needed to aid the initial design of the course. The model would need to recognise the importance of collaboration and scaffolding and be compatible with producing a course with the aforementioned characteristics of engagement and effectiveness.

5.4.1 The Synthesis of ADAPTTER

The initial prototype design model had its origin in a synthesis of collaborative unplugged activities (Chapter 3 and Chapter 5) that were placed in a structured framework influenced by Merrill's First Principles of Instruction (2002).

Unplugged activities have previously been discussed in Section 5.3.1, and are credited in their origin to Timothy Bell (Bell, Witten and Fellows, 1998). The unplugged activities (for this prototype course) were designed to be used collaboratively by students when they applied their new Computational Thinking Knowledge. The use of collaboration in this course was influenced by Vygotsky (1978), who believed that cognitive development

occurred through collaboration with an ‘expert’, be it a peer or adult (Smidt, 2009; Chalaye and Male, 2011). The other primary influencer on the initial design of the course was David Merrill, specifically his (five) First Principles of Instruction (2002).

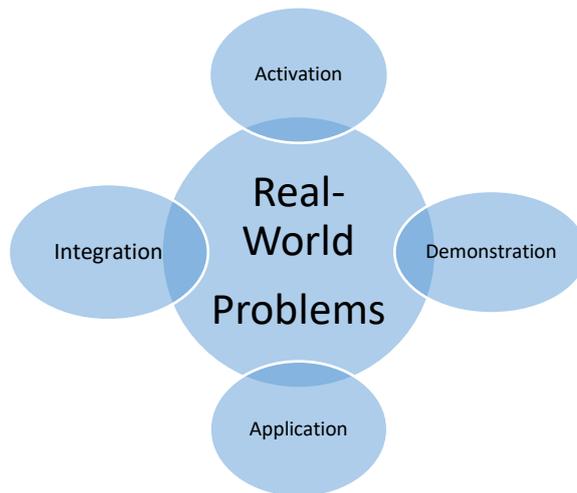


Figure 5-1 Merrill’s First Principles of Instructional Design (Merrill, 2002)

Merrill’s (2002) First Principles are concisely described as follows: learning is promoted when 1) students are involved in solving real-world problems, 2) existing knowledge is activated to support new knowledge, 3) new knowledge is demonstrated, 4) new knowledge is applied by the students, and 5) new knowledge is integrated into the student’s life. His five principles resulted from a review of pre-existing design theories, models and research, which resulted in a collection of interrelated design principles (Merrill, 2002, 2009, 2013). The aforementioned interrelated design principles became a ‘First Principle’ if it 1) promoted more effective, efficient and engaging learning, 2) could be applied to any delivery system/instructional architecture and 3) was supported by research. He defined a principle as ‘a relationship that is always true under appropriate conditions regardless of the methods or models used to implement this principle’ (Merrill, 2013, p. 19).

The rationale for using his principles to design Version 1 of the course is manifold

1. Research has shown that following Merrill’s principles results in effective, efficient and engaging learning (Merrill, 2002, 2009; Gardner, 2011; Gardner and Belland, 2012; Lo and Hew, 2017). These characteristics are important to this study’s CT course.
2. Merrill’s five principles are theoretically strong. They were derived from research across a wide selection of theories and models that ranged from objectivist to constructivist. Of note Merrill (2009) cites the following theories that support some if not all of his first principles: Andre’s (1997) Instructional Episode, Nelson’s (1999) Collaborative Problem Solving, Jonassen’s (1999) Constructivist Learning Environment, van Merriënboer’s (1997) Four Component Instructional Design Model, and Schank’s (1999) Learning by Doing model.

This study’s initial design model incorporated all of Merrill’s five principles with collaborative unplugged activities. These initial influences are still apparent in the final ADAPTTER framework, but other design components also emerged during this study.

Table 5-1 summarises the key theorists that informed or supported each component of the final ADAPTTER model. Section 5.4.1.1 provides more detail on the theoretical foundations of each component of ADAPTTER.

Table 5-1 Theoretical support for ADAPTTER

Design Component	Theoretical Foundation/Validation
Activities	Bell, Witten and Fellows (1998), Vygotsky (1978)
Demonstration	Merrill (2002)
Application	Merrill (2002)
Pre-activation	Emerged as course progressed, origins were with Merrill’s (2002) principle of activation
Theory	Cognitive Load Theory (Sweller, 1988)

Transparency	Emerged as an important design component as this study progressed. This component is validated by Rosenshine and Furst (1971)
Exemplification	Emerged as a valuable design component as this study progressed. Its origins were with Merrill's (2002) principle of demonstration and real-world examples
Reflection	Merrill (2002)

5.4.1.1 ADAPTTTER and Supporting Theorists

The design components of 'Demonstrations,' 'Application', and 'Reflection' were influenced by Merrill's (2002) first principles. Their validations as design components are discussed in Chapters 6, 7, and 8. The design component of 'Activities' was influenced by Bell, Witten and Fellows (1998), and likewise, their subsequent validation is discussed in Chapters 6, 7, and 8.

The emergence of 'Pre-activation' as an important design component is discussed in Chapter 6. This design component had its genesis with Merrill's first principle of activation. However, for this study, it was found that a baseline of knowledge had to be first created for students using memorable "pre-activation" activities, which would be referred back to later in the lesson (see Chapter 6).

The design component of 'Theory' emerged to prominence during the prototyping of Version 2 of the course by teachers (see Chapter 7). Teachers found that knowing and understanding the components of Computational Thinking facilitated their articulation of problem-solving in their other teaching subjects. This PhD course was initially designed to provide explicit teaching and guidance of new content and skills (Sweller, 1988). This study's systematic literature review highlighted the influence of Papert (1980) and constructionism in teaching Computational Thinking using programming languages. However, this view is balanced with Grover, Pea and Cooper's (2015) view that deep learning is fostered with a combination of guided discovery and instruction rather than

pure discovery. Added to this knowledge is the body of studies that recommend or suggest explicit teaching in Computer Science (Waite, 2017). Of note, Meerbaum-Salant, Armoni and Ben-Ari's (2011) study questions the use of exploratory learning with Scratch and asks if we should teach students the '“right way” from the beginning' (pg 172). Mayer (2004) and Sweller, Kirschner and Clark (2007) recommend against minimally guided teaching techniques, stating that complex concepts must be explicitly taught. This course was exploratory in nature. Academic literature provided little guidance on Irish students' pre-existing knowledge of Computational Thinking and how difficult they would find it. The initial design trialled explicit teaching to students (based on cognitive load theory (Sweller, 1988)). However, as this PhD study progressed, it was shown that due to students' lack of previous CT knowledge, explicit teaching was best placed in a framework that incorporated preactivation activities, 'explicit teaching', demonstration and application of knowledge.

Transparency was not initially considered an important design component, but its importance became prominent as this PhD study progressed (see Chapters 6, 7 and 8). This design component is validated in instructional communication research, where teacher clarity is judged as an important variable central to teaching effectiveness and learning (Qin Zhang and Baohua Huang, 2008). Teacher clarity has its origins with Rosenshine and Furst (1971), who identified clarity as one of the top five variables associated with effective teaching.

The design component of 'Exemplification' has its origins in two of Merrill's first principles, that of 'Demonstration' and 'Real-world Problems'. Chapter 7 highlights how localised, context-specific examples of Computational Thinking components were essential to CT's effective teaching and learning.

5.5 Irish Educational System

This research is undertaken within the confines of the current Irish educational system, which at the second level is dominated by the two Irish state assessments, i.e. Junior Cycle Student Award and the Leaving Certificate. There is very little time in the Irish school timetable to accommodate non-examination subjects. The one exception is Transition Year, which for Irish students is the last chance students have to engage with different topics and subjects before selecting the subjects they will undertake for their Leaving Certificate examination. Thus the intervention proposed in this research will be aimed at fifteen to sixteen-year-old students, to introduce them (which for many may be the only chance) to the problem-solving framework of Computer Science that is Computational Thinking.

5.6 What topics should the Computational Thinking course cover?

The rationale for choosing the course content is manifold: argumentative, Irish curriculum documents, research from literature, Irish policy documents, informal discussion with Irish post-primary teachers and existing websites. The Computational Thinking definition (used by this PhD study) and the Leaving Certificate Computer Science curriculum document provided the initial content categories. The topics include the components of Computational Thinking: abstraction, algorithmic thinking, decomposition, pattern matching (generalisation). Logical thinking is also included. This is mentioned in connection with CT in the Leaving Certificate Computer Science curriculum, and many academics cite its importance in connection to CT (Selby *et al.*, 2013; Csizmadia *et al.*, 2015; Curzon *et al.*, 2019). The remaining course topics were selected due to their presence on the Leaving Certificate Computer Science curriculum document. They are

ethics, artificial intelligence and search algorithms. However, these are introduced under the umbrella of CT.

5.6.1 Course Materials

Gravemeijer and Cobb (2006) describe the work of the design researcher as a bricoleur, a person who uses materials already available, adapts these materials, and, where necessary, invents new applications for these materials. This bricolage is very much evident in this study. The unplugged materials used to teach these topics were sourced and adapted from websites and books. For example, websites such as <https://www.compsci.ie/>, <https://csunplugged.org/en/>, <https://teachinglondoncomputing.org/>, <http://csunplugged.mines.edu/>, books by Hannah Fry (2018) and Paul Curzon and Peter McOwan (2017) to name a few. Care was taken to ensure the materials used were copyright-free, placed within a planned progression, and adapted where necessary. This bricolage was also ‘theory-guided’ (Gravemeijer, 1994). The emerging local instructional theory influenced the course content and structure. For this reason, the following discussion on the genesis of the course content is described in conjunction with the development of the emerging local instructional theory. (The ‘final’ local instructional theory is discussed in Chapter 9.)

5.7 Computational Thinking Course Version 1

The course consists of five units. A unit was taught during one eighty-minute lesson. The development of the content and materials, similar to the instructional design, was an iterative process. For this reason and also for clarity purposes, the content described in this chapter is the original design and contents of the course (before it was piloted). Changes that occurred to both the content and materials are described in detail in Chapter 6 and 7. A teacher’s guide was developed to capture the design and contents of the course. It contains screenshots of slides, lecture notes on same, background information

on Computational Thinking, lesson plans, links to videos, design layout of each lesson and the instructions on how to use the unplugged activities.

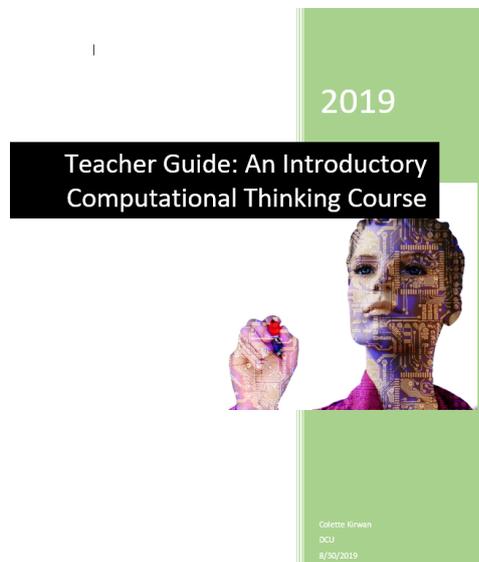


Figure 5-2 Cover of the Teacher Guide

5.7.1 Unit 1: Problem Solving and Computational Thinking

Unit one is concerned with problem-solving, specifically generic problem solving and its subsequent relationship to the problem-solving framework of Computer Science, i.e. Computational Thinking. The four pillars (components) of Computational Thinking are introduced, with three of them (decomposition, abstraction and pattern matching) being discussed in detail 5.7.1.3.

5.7.1.1 Student Learning Outcomes

- Critically assess problem-solving using both a strategy and guesswork.
- Identify the components of the Computational Thinking framework.
- Apply elements of the Computational Thinking framework to help solve a problem.

The learning outcomes were constructed based on Bloom’s taxonomy with Bowe and Fitzmaurice’s (2011) guide used as a guideline.

5.7.1.2 Materials Used (low resource)

Type	Reference
------	-----------

Slides	Slides_CT_Week1.pptx
Videos	Fellowship of the Ring Storyboard Part 1 The genius of the London Underground (Ted Talk)
Documents	ECT Lesson Plan: Finding Patterns in Spelling Errors and History, Pencil_code_emoji

5.7.1.3 Content Rationale

The Computational Thinking framework is introduced to students gradually. A generic framework for problem-solving, Polya’s (1945) framework, is first discussed. This leads to the introduction of the problem-solving framework of Computer Science, Computational Thinking.

The CT concepts are first introduced individually before they are collectively used to solve a problem. Three components are presented in this unit: decomposition, abstraction, pattern matching. These CT concepts are introduced under the guise of the four techniques/pillars idea. This idea is used on the BBC BiteSizes website (BBC, 2020), the ‘Computational Thinking for Problem Solving’ MOOC (Davidson and Murphy, 2018) and also during the Leaving Certificate Computer Science National Workshop 3 (PDST and DES, 2019b). Algorithms are introduced with algorithmic and logical thinking in Unit 2.

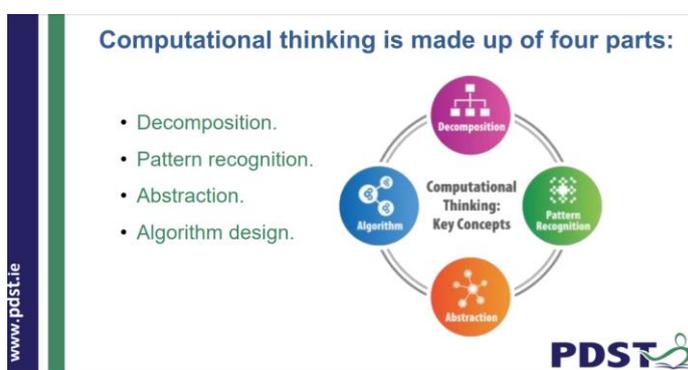


Figure 5-3 Slide taken from the PDST Leaving Certificate National Workshop 3, Session 3 (2019)

The definition used for each of these four concepts of Computational Thinking are:

- **Decomposition:** Break a complex problem into smaller, manageable subproblems.
- **Pattern Matching:** Identifying patterns and regularities in data. This can help with understanding the problem and also with devising a solution.
- **Abstraction:** Identify and capture the essential details. Ignore unnecessary detail.
- **Algorithms:** An ordered series of instructions for solving a problem.

The above definitions were influenced by the Google Computational Thinking Concept document (Google, 2018), Irish educational Curriculum documents (NCCA, 2016; NCCA and DES, 2018) and books and websites that implement the UK's GCSE Computer Science curriculum (Lawrey, 2018; BBC, 2020). The Google Computational Thinking Concept document was subsequently referenced by the PDST at the 4th National Workshop for the Leaving Certificate Computer Science course for Computational Thinking (PDST and DES, 2019a). The definitions used in this PhD course are purposely generic so that they can apply to disciplines outside of Computer Science; for example, the definition of abstraction and algorithms do not mention programming or computers. This contrasts with Denning and Tedre's (2019) definition of algorithms that refer to the steps being performed on a machine. Regarding abstraction, it is the reductionist and simplification aspects of abstraction as specified by Wing (2009) that is emphasised. There are two types of abstraction in Computer Science, modelling and encapsulation (The Open University, 2019). Wing (2009) emphasises the modelling approach, i.e. ignoring detail that is not of interest. The encapsulation approach is concerned with information hiding, which is very evident in programming. As most students had not programmed, this type of abstraction is not referenced in the course. It should be noted that my original definition for abstraction was 'Abstraction: Identify and capture the essential details. Remove un-necessary detail'. This was influenced by the Irish Primary Maths background document's definition which stated: 'abstractions (removing unnecessary detail)' (NCCA, 2016, p. 27) and also the GSCE abstraction definition on the

BBC Bitsize site (BBC, 2021). My definition now reads ‘Identify and capture the essential details. Ignore un-necessary detail’. This change was requested by Expert 4 (see 8.5.4).

5.7.1.4 *Instructional Design in Unit One*

The following practices and principles were used in the design of the instruction:

- real-world problems
- unplugged activities
- activation
- reflection (integration)
- application
- demonstration
- collaboration

Merrill’s first principles combined with unplugged activities formed the initial design characteristics.

Unit One starts with two logic puzzles. Their purpose is two-fold, 1) as a warm-up engagement activity (Mingguang, 1999; Byrne, Kearney and Sullivan, 2018) and 2) to highlight that a strategy for problem-solving is better than guesswork. An example of one of the puzzles is as follows:

Jack is looking at Anne, but Anne is looking at George. Jack is married, but George is not. Is a married person looking at an unmarried person? A) Yes B) No C) Cannot be determined. (Hector Levesque cited by Stanovich, 2014)

The puzzles are specifically chosen as research shows 80% of participants get the wrong answer (Stanovich, 2014). This is cognitive laziness; participants don’t use a strategy or framework when solving the above puzzle; they make an assumption and choose C. (The correct answer is A.)

5.7.1.5 *Reflection and Theory*

Students are next introduced to the Computational Thinking framework using an incremental approach. They reflect on the previous logic puzzles and use this experience to assess the advantages of using a framework to solve a problem. They are first shown Polya's framework and then shown the framework used in Computer Science, i.e. Computational Thinking (Polya, 1945). They are next asked to think about problem-solving in their own lives.

5.7.1.6 *Activation and Demonstrations:*

Individuals then discuss the four pillars/techniques. Students are first asked to provide examples, before exemplars are provided from Computer Science and everyday life. The purpose is to activate previous knowledge.

The CT components are next demonstrated with the help of short videos and code demonstrations. For example, a video of the storyboards used in 'The Lord of the Rings: Fellowship of the Ring' film illustrates decomposition (Jackson, 2001). Storyboards used in mobile app development are shown next. Abstraction is demonstrated with a video of a Ted Talk on the London underground. Similarly, the smiling emoji is used to highlight pattern matching, i.e., three circles and a semi-circle. The code that creates the Emoji (shown in Figure 5-4) is run on the pencilcode.net site to show how lines of code are repeated to draw circles of different shapes. The smiling emoji is used again in Unit 2. Exemplars were also used from everyday life, examples from abstract art (abstraction), writing an essay (decomposition), models such as a periodic table, water cycle and fingers prints (pattern matching). (Note most of the exemplars containing non-computer elements did not make it to Version 2.)

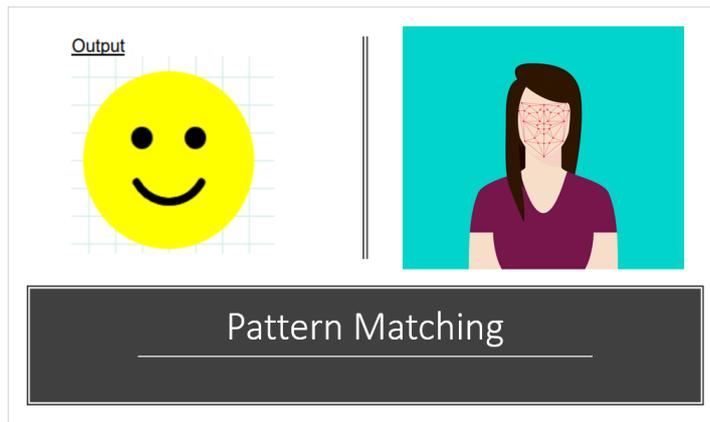


Figure 5-4 Slide used to highlight pattern matching in Computer Science

5.7.1.7 Application

Students are next asked to apply their knowledge of Computational Thinking to a scaffolded group activity. The purpose of the activity is to use Computational Thinking to design a spellchecker. It is inspired by a lesson plan from the CT for Educators course ‘CT Lesson Plan: Finding Patterns in Spelling Errors and History’ (Google, 2015).

Students are first asked to decompose the problem (they discuss how to check and correct a word for mis-spelling). They are next given a spelling test. The test consists of six of the most misspelt words in the English language: accommodate, separate, definitely, pharaoh, occurred and until (Curtin, 2018). Students’ answers and the most common mis-spelling of these words (these were usually the same) and the correct answer were written on the white/blackboard. In groups, they were asked to identify any patterns in the mis-spellings, and from there, to come up with an algorithm to design a spell checker similar to that observed in MS word. Students share their answers and reflect on the process.

They are then shown how one such solution can be performed by a computer, thus taking advantage of the computer’s computational power and speed. The code for this solution was modified from Peter Kuhar Java solution on GitHub, who subsequently modified his code from Peter Norvig’s python code (Sentance, 2014; Kuhar, 2016; Norvig, 2016). The computer program checks each inputted word in the sentence against a ‘dictionary’ to see

if the word is spelt correctly. It does this fast. If the word is not there, it then uses the ‘patterns’ that were previously identified in the mis-spellings activity to check if that ‘new’ word exists in the dictionary; it then corrects the spelling.

Chapter 6 and 7 document the many changes that occurred both to the content and the course design for Unit 1.

5.7.2 Unit 2: Algorithms and Logic:

The purpose of this unit is to introduce students to the last core concept of Computational Thinking, i.e. algorithms, specifically their design and characteristics. Logic also forms a core element of this lesson. This unit has five learning outcomes. At the end of the lesson, students should be able to:

- Explain the importance of Algorithms in the Computational Thinking framework.
- Write unambiguous, consistent algorithms.
- Appreciate how logic is used in algorithms to ensure validity.
- Appreciate and discuss how computers do not think for themselves. They are an automation of our reasoning.

The lesson will also help students to:

- Develop their logical thinking.

5.7.2.1 Materials Used

The materials needed for this class are: chocolates, a chilli, a glass jar (to hold chocolates and chilli), playing cards and the Codebreaker worksheets.

Type	Location
Slides	<ul style="list-style-type: none"> • Slides_CT_Week2.pptx
Videos	<ul style="list-style-type: none"> • Screencast of spellchecker—spellchecker.mp4
WebSites	<ul style="list-style-type: none"> • Codebreaker AI (online Codebreaker game)
Documents	<ul style="list-style-type: none"> • Codebreaker worksheet

Materials	<ul style="list-style-type: none"> • Chocolates, a chilli, a glass jar (to hold chocolates and chilli), • playing cards
-----------	---

5.7.2.2 Content Rationale

As stated earlier, the definition used for algorithms is based on generic characteristics. In this study’s course, algorithms are defined as an ordered series of instructions for solving a problem, i.e. the step-by-step written solution. They are defined as having the following characteristics.

- They consist of a series of steps that are precisely defined.
- The steps are sequential and can be considered ‘an instance in time,’ i.e. once a step is executed, it is forgotten (Beecher, 2017).
- The steps are consistent, i.e., based on their input, you should know the output.
- The steps are defined unambiguously.
- The user (i.e., human or computer) executing the steps does not need to understand them, to follow them. (Although in the ethics section, we discuss the need for the transparency of algorithms for humans. This is now very topical with predicated grades.)

For the purpose of this study’s course, an algorithm can be a cooking recipe, Lego instructions or a computer program, e.g. a mobile application. As stated earlier, this is different from other definitions, which refer to the steps being performed on a machine, but is in line with the Irish Leaving Certificate Computer Science Specification: ‘An algorithm is a sequence of steps designed to accomplish a specific task. Algorithms are translated into programs, or code, to provide instructions for computing devices’(NCCA and DES, 2018, p. 32).

Logical thinking is introduced due to its relationship with algorithms. To explain further, algorithms build on logic; in short, when you write algorithms, you make logical decisions (Beecher, 2017). In fact, logic is crucial to CT (Beecher, 2017, p. 15). It is

logical thinking that helps to ensure the validity of our reasoning. This unit aims to highlight to students that computers do not think for themselves. They are an automation of our reasoning, which should be valid.

5.7.2.3 Instructional Design

Similar to Unit One, the following processes were used in the design, (pre) activation, integration, demonstration, application and real-life problems (to a lesser extent). These principles were applied using collaborative and individual unplugged activities. The unplugged activities used were the chilli chocolate game, a magic card trick, Codebreaker game, and creating an algorithm to draw an emoji. The design of this unit changed before it was piloted. This was due to feedback from the piloting of Unit 1. The unplugged activities (chilli chocolate game and card tricks) were moved to the start of the unit. This was done to establish a common baseline of knowledge before the corresponding content on algorithms was introduced. The characteristics of an algorithm are then explained after the activities.

5.7.2.4 Pre-Activation

The unit starts with a recap of the definitions of CT and its four main techniques. This is followed by two activation activities, the chilli-chocolate game and an individual card-trick. The purpose of these activities was now manifold, 1) engagement/warm-up, 2) act as a simplified model, 3) to learn the characteristic of algorithms by doing, before the theory is introduced.

The chilli-chocolate game involves thirteen chocolates and one chilli. Two people play the game. Each person, in turn, can take one, two or three chocolates out of the jar, with the objective of being the person not left with the chilli pepper at the end. This is a Nim type game sourced from the BBC documentary ‘The Secret Rules of Modern Living Algorithms’ hosted by Oxford Mathematician Professor Marcus du Sautoy (Briggs,

2017). The lead-in to the game is such that the teacher challenges a volunteer student to play, explaining that (s)he will play the game using an algorithm. The student is challenged to beat the teacher or else discover how the algorithm works.

The second activity is a self-working card trick. Paul Curzon's work on magic as a pedagogy to teach Computer Science inspired the use of card tricks to explain algorithms (Curzon and McOwan, 2008, 2013, 2017). The trick is called the Royal Family or Jack, King, Queen and Ace's trick. It is demoed on YouTube (Nemzer, 2011) The trick is first demoed to students. Its 'workings' are not explained. Each student is given a deck of cards and asked to perform the same trick by following the teacher's instructions. The purpose of this magic trick activity is to highlight to students 1) the consistency of algorithms, 2) that computers blindly follow instructions, and 3) to emphasise that you don't need to know the workings of an algorithm to use it.

5.7.2.5 Reflection and theory

When completed, students reflect on the activities and use this new knowledge to explore the characteristics of algorithms.

5.7.2.6 Demonstration and Application

Students now write their own algorithms. The lead-in to this activity involves watching a two-minute video titled the 'Exact Instructions Challenge' video (Darnit, 2017). A dad has challenged his kids to write the instructions for a peanut jelly sandwich. This video demos to students the consequences of not having clear and precise instructions. In groups, students write explicit instructions (algorithms) on how to draw the following emoji (see Figure 5-5).

Output

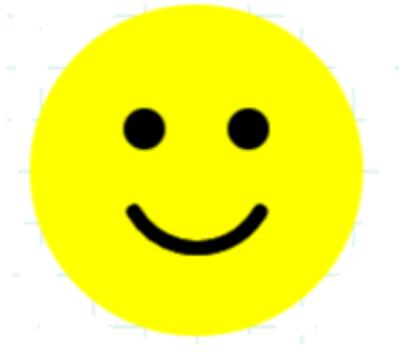


Figure 5-5 The emoji that students re-create using instructions.

5.7.2.7 Reflection

Student algorithms are tested on other groups and the teacher, to see if a similar emoji can be replicated.

5.7.2.8 Warm up Puzzle

Students are given a short break before the topic of logic is introduced. Logic is first presented using a puzzle. Its purpose is two-fold in that it serves to engage students whilst also introducing the topic of logic. The lead-in to the puzzle is presented using a video clip from Die Hard 3. There is a bomb planted at a fountain. It can only be de-activated by placing 4 litres on a scale. However, you only have two containers available to use, a 3-litre and a 5-litre. In groups, students write out the algorithm to solve the puzzle. (Note, the puzzle and video were introduced based on feedback from Unit 1 and Unit 2.)

5.7.2.9 Reflection

The teacher tests each group's results, and together they discuss their answers.

Algorithms need to be accurate and concise, but are they valid? Does the chain of reasoning lead to a conclusion, i.e. four litres in a container? This leads to a brief introduction of logic, where deductive and inductive logic is briefly explained. Students will now further develop their logic skills to 'break the code'.

5.7.2.10 Demonstration and Application

Mastermind or Codebreaker is a game that has been shown to improve logical thinking (Wood, 1980). A summary of the game is as follows, one student (code maker) picks a code, and the other student (code breaker) must break it. The secret code contains four colours, out of a choice of six colours. The code breaker then tries to guess the code. The code maker provides feedback on the guess. The students engage in logical reasoning as they need to interpret the feedback provided by the codemaker and use this information to take a 'better' guess which is reducing the number of possible solutions (Strom and Barolo, 2011). They go from the general to the specific. There are 1,296 possible solutions, and the students will try to guess the code within ten guesses using logic. The lead up to this activity involves demoing the game first online using the following website <http://csunplugged.mines.edu/codebreaker/game.cgi>.

5.7.2.11 Reflection

Students reflect on their activities and discuss the strategies they used. This unit finishes by looking at how you could break the code using the computational power of a computer. The computer can employ exhaustive search, i.e. generate all possible combinations (1,296) then eliminate values quickly based on the feedback from the guess. The course returns to logic and algorithms when we look at pseudocode.

5.7.3 Unit 3: Common Algorithms:

In this unit, students use 'bread and butter' algorithms alongside decomposition, pattern matching and abstraction to solve a real-world problem. The algorithms introduced are tried and tested techniques to solve problems specifically around searching, i.e. linear and binary searching. These algorithms form part of the Leaving Certificate Computer Science curriculum. This unit has four learning outcomes. At the end of the unit students should be able to:

- Differentiate between Linear and Binary search
- Evaluate an algorithm
- Assess an algorithm
- Apply Computational Thinking skills to solve a problem

5.7.3.1 Materials Used

The materials used are worksheets for the activity Searching to Speak.

5.7.3.2 Instructional Design

Similar to Unit 1 and 2, the processes of activation, demonstration, application, and unplugged activities occur but in a different order to the previous units. The design of this unit changed dramatically before its piloting, based on feedback from the piloting of Unit 1 and Unit 2. A warm up exercise was added, and the binary search algorithms were discussed with students first attempt at the Searching to Speak activity

5.7.3.3 Warm up Puzzle

The course starts with a recap of the definitions of CT and its four main techniques. A collaborative engagement activity follows this, i.e. a logic puzzle, which also serves to (re)introduce the concepts of decomposition (divide and conquer) and elimination which will be explored more when we do a binary search.

The puzzle was sourced from Khan Academy. It is as follows, ‘You have nine identical balls. One of them is slightly heavier than the others. Using a balance scale, find this ball. What is the fewest numbers of times you have to use the scale to find the heavier ball?’ (Khan Academy, 2014). Students complete the puzzles in groups and discuss their answers.

5.7.3.4 Application

Students now divide into groups to complete a scaffolded Computational Thinking exercise. The activity is sourced from Paul Curzon’s ‘Computer Science for Fun’ (CS4fn) website and is also available in his book (Curzon, 2014a; Curzon and McOwan, 2017).

The activity is inspired by Jean Dominique Bauby, who on December 8th 1995 suffered a major stroke. At the time he was the editor of the fashion magazine Elle. He woke up twenty days later to discover he had locked-in syndrome. His mental faculties were intact, but most of his body was paralysed, i.e. mouth, arms, legs. He was unable to speak; all he could do was blink his left eyelid. Despite his situation, he wrote a best-selling novel, ‘The Diving Bell and the Butterfly’. He composed, edited and dictated the entire book, in his head one letter at a time. In groups, students are tasked to use Computational Thinking to devise a plan that would provide a solution that would allow a person in a similar situation to Bauby to communicate. Their solutions are then used to ascertain their group members’ middle names. Templates are provided to help scaffold the activity.

5.7.3.5 Reflection

Students’ communication protocols are shared with the class. Group discussions are facilitated, with prompt questions:- Were they successful, could they write down the middle name of a partner by only following their rules? How did they turn blinks into letters? How long did it take?

5.7.3.6 Theory and Demonstration

Students now investigate if they can improve their solutions by using some bread and butter algorithms from Computer Science, such as searching algorithms: linear search, and binary search. The linear search algorithm is discussed in relation to searching for a book in an unordered library. The binary search algorithm is discussed using a phonebook. This idea was sourced from Harvard professor David J Malan course ‘Introduction to Computer Science’ (15:30 to 17:10), available on YouTube (Malan, 2014). Binary Search is demoed by ripping up a phonebook. For example, if a phone book has 1,000 pages, and you are looking for a specific name, Sullivan. Open the book in the middle, check the name at the top. Next, rip the book in two, and discard the half

that is not relevant. You have just got rid of 500 wrong pages. You repeat the process.

The CT concepts of decomposition and pattern matching are evident in the working of a binary search algorithm (halved the size of the problem with each step, pattern matching: repeat the steps).

5.7.3.7 Application

Students return to the Search to Speak activity and thus write a new protocol using binary search.

5.7.3.8 Reflection and Real-World Problem

Students reflect on the use of binary search.

5.7.4 Unit 4: Ethics and Artificial Intelligence (AI)

Week 4 introduces students to two topics: artificial intelligence and ethics. This is achieved by first looking at artificial intelligence in ‘Real Life’ for example, Google Assistant, smart emails, Netflix and Spotify. The field of ethics is explored from the standpoint of both the creators and users of technology. Using thought experiments and examples from real life, the decision-making process of programmers is explored. A goal for this lesson is to highlight how 1) ethical thinking is essential in Computational Thinking and that 2) whilst AI machines, are very much a part of our lives, they are not self-aware, a human writes their code.

5.7.4.1 Content Overview and Rationale

The inspiration for this topic was the Leaving Certificate Computer Society ‘Computers and Society’ strand and Curzon and McOwan (2017) comment in relation to creating ChatBots ‘we will see why computational thinkers, whether human or machine, need ethical thinking too’ (Curzon and McOwan, 2017, p. 115). The first half of this lesson is concerned with artificial intelligence and its relationship to Computational Thinking. The goal here is to 1) introduce students to artificial intelligence, 2) explain how ‘these programs’ are very much a part of their lives, 3) make students aware of the ethics

associated with these programs, and 4) to emphasise that ethical thinking is important in AI because these programs are still written by humans (the machine is not intelligent; they are not creative; they do not have free-will). AI is introduced to students using a chatbot. A scripted chatbot is first demoed, students next use CT to create their own chatbot. An AI chatbot is next demoed, and its ethics explored. AI examples in real life are also introduced. This topic finishes by explaining the algorithms behind AI programs.

The second part of this lesson focuses on moral and thought experiments which are used as prompts to discuss ethics in Computational Thinking. The rationale for using this topic is as follows: it relates to the Leaving Certificate curriculum component: computers in society, it is different hence will perhaps interest students who are not interested in Computer Science, and more importantly it highlights the importance of algorithm accountability and the fact that our Computational Thinking should be both valid and morally correct (Curzon and McOwan, 2017; Fry, 2018; Shah, 2018).

This unit had three learning outcomes. At the end of the unit, students should be able to:

- Discuss the importance of Ethical Thinking when programming
- Explain Artificial Intelligence
- Explore ethical dilemmas that impact us as both users of technology and also developers of technology

Although the learning outcomes for this unit changed as the course progressed, see Chapter 7.

5.7.4.2 *Material Used*

Type	Location
YouTube	‘Google’s AI Assistant Can Now Make Real Phone Calls’ Telco chat bot sample

	The Man Who Saved the World Trailer 1 (2015) – Stanislav Petrov, Kevin Costner Documentary HD
WebSites	20q.net. The Neural Network on the internet Moral Machine ELIZA
Word Document	ChatbotInstructions.doc ECT Pencil Code Program: Lady Macbeth Chat Bot

5.7.4.3 Instructional Design

Once again, the principles and techniques used in previous lessons are used, such as pre-activation, demonstration, application, reflection and unplugged activities. In this instance, the activities are very much discussion and reflection based, rather than practical and hands-on. Videos and online applications play a very important role in this lesson. The following section describes the design and content that was piloted for Version 1. Similar to before, a pre-activation step was added to Version 1, based on feedback from units 1, 2, 3. However, the design and focus of this unit were significantly changed for Version 2 of the course (see Chapters 6 and 7).

5.7.4.4 Demonstration and Application

The lesson starts with a video of the google duplex AI assistant, which is capable of holding a two-way conversation with an employee at a hair salon and a small restaurant (Mashable Deals, 2018). This video serves as an introduction to AI and a baseline to highlight its difference from scripted chatbots (when they are introduced). Definitions related to AI are next discussed, such as bots, machine learning etc. Notably, a formal definition of AI was never stated in Version 1.

5.7.4.5 Demonstration to explain concepts

To reinforce that AI is under the control of algorithms, two algorithmic types (neural networks and random forest) are discussed using both exemplification and

demonstrations. The neural network algorithm is demonstrated with a teacher-led activity. Students, together with their teachers, play an online game of twenty questions, <http://20q.net/>. The 20AQI is a neural network-based artificial intelligence programme. When you play the game, the AI learns by repetition. The program has a goal: guess the user's object; the user gives it feedback as to how to direct its questions and if the final guess is right/wrong—the contradictions section illustrates (see Figure 5-6) how the program learns from feedback.

20Q won!

Play Again

You were thinking of tea.
Do you know any songs about it? 20Q was taught by other players that the answer is No.
Does it contain a liquid? You said Yes, 20Q was taught by other players that the answer is Doubtful.

Contradictions Detected
The opinions of the 20Q A.I. are its own, and are based on the input of players. 20Q's answers reflect common knowledge. If you feel that 20Q is in error, the only way to correct it is to play again!

Similar Objects
green tea, coffee, honey, a cucumber, saliva, milk, ice, a hard-boiled egg, a weed (undesirable plant), a tomato plant, a cherry tomato, food.

Play Again

How 20Q learns

Figure 5-6 Screenshot of the <http://20q.net/> site.

5.7.4.6 Real-Life Examples and Reflection

Students next discuss ‘real-world examples’ of A.I. Prompts such as Google Assistant, Gmail: Smart Reply, Spotify, Netflix are used in the discussion. Students also discuss ethics and AI, with the chatbot being used as a catalyst for discussion. Chatbots have many practical uses: virtual worlds in games, helpdesks on websites. On the dark side, bots can spread lies with the purpose of affecting people’s views and opinions (Fry, 2018). Similarly, one can consider the effect of bad data on AI programs; a pertinent example is the AI chatbot TAY (Thinking About You). This bot was released via Twitter on March 23rd, 2016 by Microsoft. It was shut down sixteen hours later, as it began to post racist, misogynistic, offensive tweets. It had been targeted by trolls, who tweeted sexist, racist tweets to the bot, which ‘learned’ from these responses(‘Tay (bot)’, 2019).

5.7.4.7 Demonstration and Application

Students next use Computational Thinking to create their own chatbot. This activity is based on the chatbot activity written by Curzon and McOwen (2017). The goal for this group activity is to create a believable ‘scripted’ conversation using Computational Thinking. Before they create their chatbot, the ELIZA chatbot (Weizenbaum, 1966; Berkers, 2003) and a ‘pencil code’ scripted chatbot is first demonstrated to students. This demonstrates how a chatbot identifies keywords in a ‘user’s’ conversation and then matches them with the ‘computer’s’ scripted responses (Decomposition and Pattern Matching) (McKnight and Google Exploring Computational Thinking Team, 2015).

5.7.4.8 Reflection

Students reflect on their chatbots, using the following prompt questions. Was their chatbot realistic? Were they able to have a believable conversation? Would their chatbot pass the Turing test? How would they rate it, on a scale from 1 to 5? What were the strengths and weaknesses of their chatbot? What needs to be improved? What gave it away, as being ‘fake’? (Curzon and McOwan, 2017, pp. 88–91).

5.7.4.9 Ethical Thinking CT

The second part of this lesson focuses on ethics, specifically ethical thinking being a part of Computational Thinking and thus technology.

5.7.4.10 Pre-activation Warm-up puzzle

The ethics part of this lesson starts with a puzzle, i.e. a philosophical thought experiment. Its purpose is manifold, engagement, and a ‘pre-activation’ activity, as it relates to driverless cars, which are discussed later in this lesson. (This warm-up puzzle was added based on feedback from Unit 1, 2 and 3). The thought experiment is as follows: You see a runaway trolley car. It is moving towards five people who happened to be tied to the track. They will die if the trolley hits them. But there is a lever close to you. It controls a

switch that will redirect the train to a different track, where there is only one person. What do you do?

Do nothing and watch/allow five people to die.

Pull the lever and kill one person.

The definition of ethics that the course subscribes to is: ‘At its simplest, ethics is a system of moral principles. They affect how people make decisions and lead their lives’ (BBC, 2014, para 1).

5.7.4.11 Demonstration: Video Driverless Cars and Activity

To help a student understand exactly what a driverless car entails, students are shown a video of a driverless car in Las Vegas (Brownlee, 2019). This video serves as a catalyst for student discussion:- they are asked to compare their answer to the trolley problem with their answer to the following question: ‘Would you buy a driverless car that would prioritise your life in the event of a crash or one that would react to save as many lives as possible’ (Fry, 2018).

5.7.4.12 Activity

Students are next tasked with discussing in groups the ethics surrounding driverless cars.

An activity sheet is provided that poses certain scenarios taken from MIT’s moral machine, accessed at <http://moralmachine.mit.edu>.

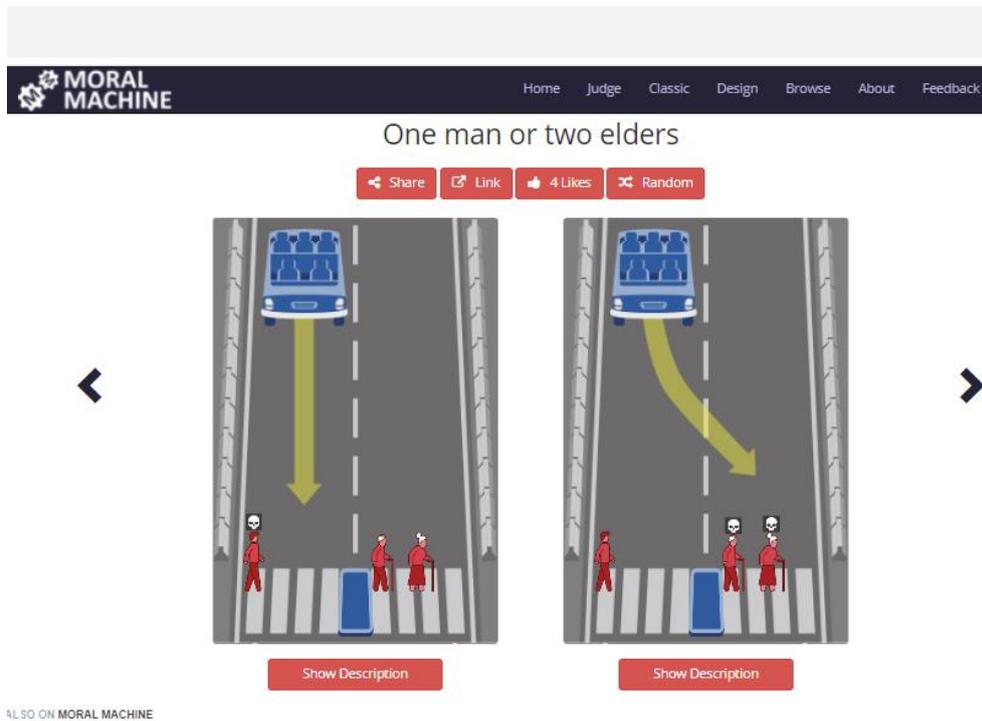


Figure 5-7 Screenshot of Moral Machine (Scalable Cooperation and MIT Media Lab, no date)

5.7.4.13 Discussion activity using real-world examples and video demonstrations

A class discussion follows the Ethics worksheet. Using real-world examples, students are tasked with discussing ethical dilemmas related to the decision making processes of computers.

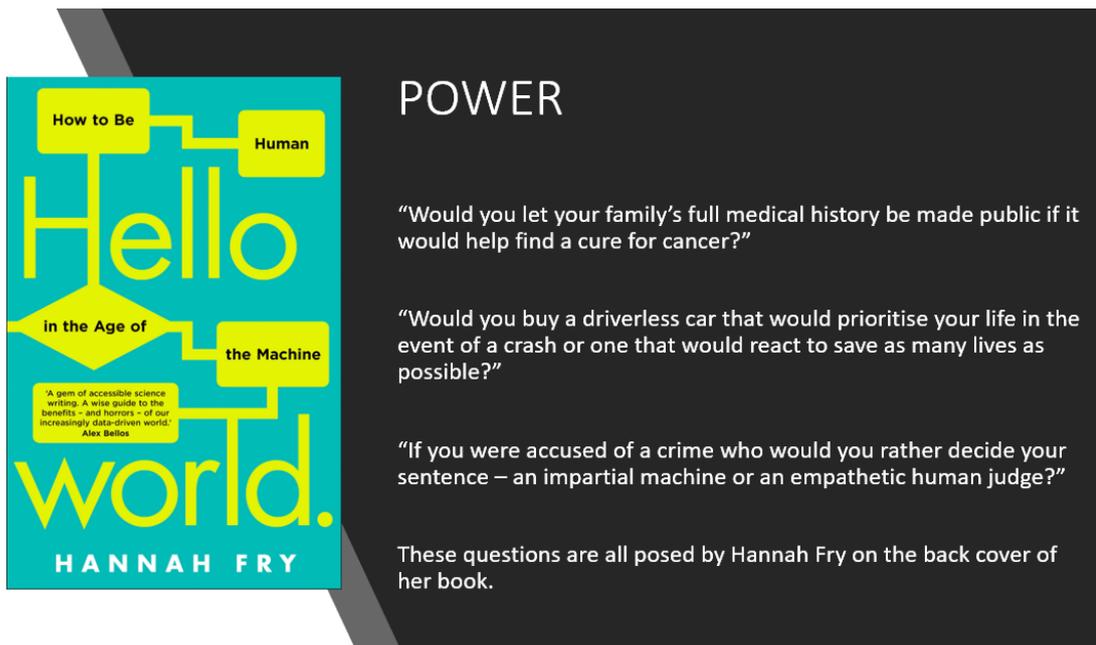


Figure 5-8 A slide showing a picture of the front cover of Hannah Fry's book with the questions on the back cover (Fry, 2018).

Power: Students are played the trailer from the documentary ‘The man who saved the world’. This film concerns Stanislav Petrov, a lieutenant colonel in the Soviet Air Defence Forces. On September 26, 1983, he chose to ignore the computer warnings that five US nuclear warheads were heading for Moscow and thus is credited with preventing a nuclear war. Students are asked in their groups to consider what they would have done in Stanislav’s place. Justice: Students are next asked the following question: ‘If you were accused of a crime who would you rather decided your sentence – an impartial machine or an empathetic human judge?’ (Fry, 2018). Health: ‘Would you let your family’s full medical history be made public if it would help find a cure for cancer?’ (Fry, 2018).

5.7.5 Unit 5: Fundamentals of Programming

The goal for this unit is: 1) to link Computational Thinking to programming, 2) to map algorithms (written in English) to pseudocode and to introduce students to data representation.

5.7.5.1 Content Overview and Rationale

The unit adopts the approach that there are five fundamentals to programming. If you follow these rules, you can write algorithms in any programming language. The rules are input, output, sequence, repetition and decisions. These fundamentals are drawn from my own experience of teaching Java and Android Development. This unit had three learning outcomes:

- Describe the fundamentals of programming
- Explain how CT relates to programming
- Write pseudocode.

5.7.5.2 Instructional Design

Similar to other units, the following processes were used in the design: activation, integration, demonstration, application, unplugged activities and real-life problems. The following section describes the design and content that was piloted for Version 1. However, the design and focus of this unit were significantly changed for the piloting of Version 2. They were also changed again during the pilot of Version 2 ('on the fly'). The biggest change was that the language Python was introduced as an alternative to pseudocode (see 7.1.4).

5.7.5.3 Activation and Theory

The unit starts with a recap of CT components. An activity for writing an algorithm for making a cup of tea follows. The purpose was to re-activate previous learning and to make explicit the connection between everyday algorithms and computer programmes. Students answers were discussed with the purpose of highlighting the fundamentals of programming.

5.7.5.4 Demonstration:

The teacher demonstrates how the identified fundamentals can be turned from an English algorithm into pseudocode algorithm (decision statements become IF statements, repetition statements become While statements).

5.7.5.5 Demonstration

The teacher next demonstrates to students how a computer ‘thinks in twos’. Using a balance scale and a group of items (of different weights), students are ‘visually’ shown how a computer program calculates the maximum or minimum number in a list. Two items are put on each side of the balance scale, the heaviest item would remain, and the next item would be placed on the scale. The heaviest item at the end is the heaviest (max) number in the list.

5.7.5.6 Application

In groups, students write out the algorithms in English and then try to identify the pseudocode that would correspond to their answers. A template and a cheat sheet for this task are provided. A snippet of the cheat sheet is displayed below.

Cheat Sheet		
Concept	Example Pseudocode	Notes
Variables	<pre>Counter = 0 Max = 0 name="Colette" myList = {2,6,1,6,5,9}</pre>	INPUT OUTPUT Variables are assigned using the = operator
Outputting to screen	<pre>PRINT("hello") PRINT ("Enter a grade")</pre>	PRINT(string)
Taking input from user	<pre>name = INPUT("Please enter your name") myList = INPUT("Please enter a list of numbers")</pre>	Variable = INPUT(prompt to user)
Selection	<pre>IF entry == "a" THEN PRINT("You selected A") ELSEIF entry == "b" then PRINT("You selected B") ELSE PRINT("Unrecognised selection") ENDIF If max < myList[i] Then max = myList[i]</pre>	IF / ELSE selection

Figure 5-9 A snippet of cheat sheet used in Week 5. Cheatsheet adapted from OCR GCSE in Computer Science Specification (Oxford Cambridge RSA (OCR), 2020)

5.7.5.7 Reflection

Students and teachers together reflect on their answers.

5.8 Summary

This chapter presented definitions for key components of this research: engagement, quality, low threshold, unplugged activities, and effectiveness. It highlighted a gap in the literature regarding how teachers use unplugged activities to teach Computational Thinking. The rationale for the initial design and contents of the course was presented, with Davis Merrill's principles introduced. An overview of the initial design and contents of the course was presented. The revisions and changes that occurred to this initial design are discussed in Chapter 6 and Chapter 7. The following chapter will describe the

prototype phase of this methodology in detail, specifically the prototyping of Version 1 of the intervention.

6 Prototype Phase Version 1

6.1 Introduction

This chapter concerns the prototype phase of this study, specifically the prototyping of Version 1 of the Computational Thinking course. It presents a detailed narrative on the implementation and evaluation of Version 1. The discussions are context-rich, they describe the participants, the setting, the course revisions, design decisions, and research results. This is recommended as a measure to guard academic rigour (McKenney, Nieveen and Van den Akker, 2006). The chapter is concerned with the following topics: the evolution of the research design, recruitment, qualitative and quantitative data analysis, findings, and recommendations to the instructional design and content. It also concerns evaluations, as the course is evaluated to ascertain whether it is effective, high-quality, practical, and engaging.

This prototype phase was conducted over nine months, from September 2018 to May 2019. It involved eighty-six students, five teachers, three content experts and four schools.

6.2 Research Design Evolution

This section starts with an overview of the research approach used to design, implement and evaluate the Computational Thinking course (the intervention). While this chapter is concerned with Version 1 of the course, the overview of the research design provides a summary of all versions. This is necessary to add context and clarity to Version 1.

Similar to the design of the intervention, the research approach also underwent iterative cycles. It changed to meet both the needs of the teachers and the research. Figure 6-1 illustrates the actual approach taken to complete this study.

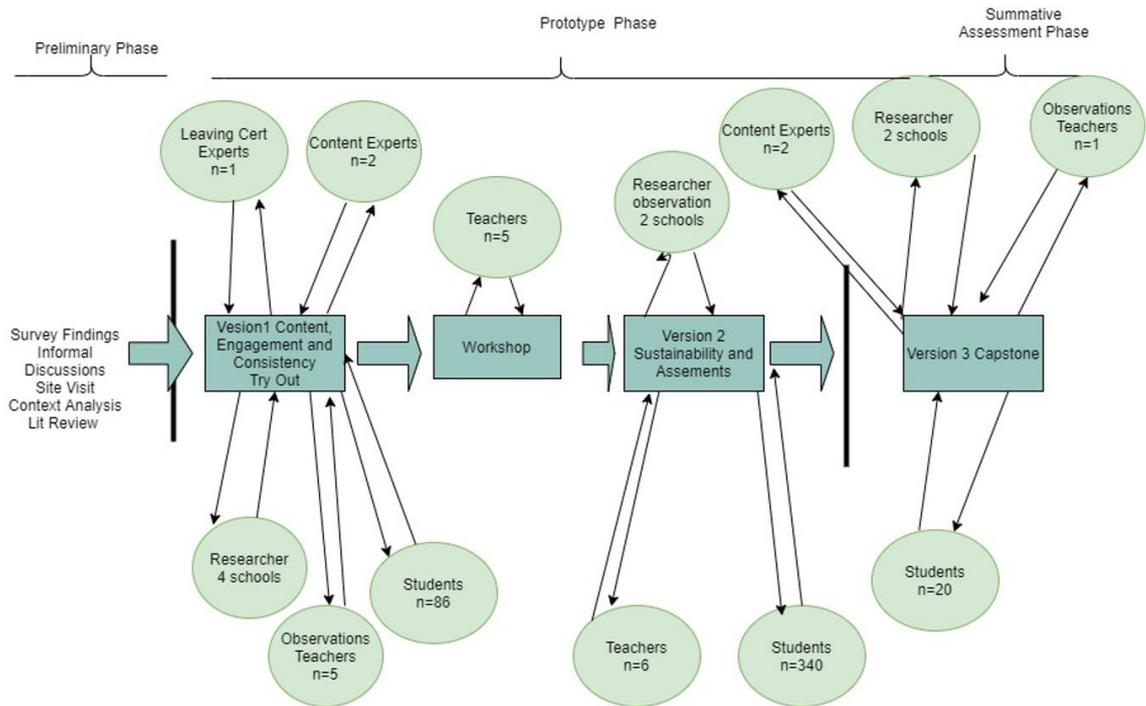


Figure 6-1 Research Approach

This contrasts with Figure 6-2, which shows the originally proposed design path.

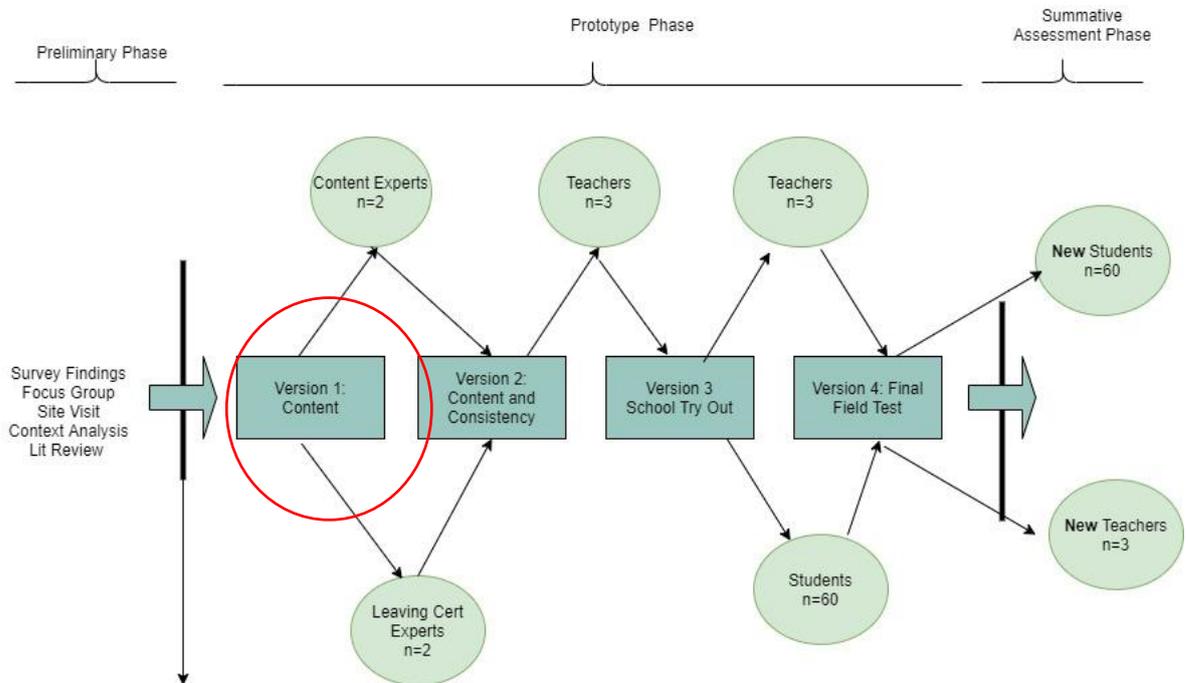


Figure 6-2 Original Design Approach

The original approach was developed based on 1) feedback from the preliminary analysis phase of the study, 2) informal discussions with post-primary teachers and 3)

consultations with researchers at DCU. It can be understood as having the following requirements and milestones (see Figure 6-2). Version 1 would be developed based on pre-existing unplugged materials, (Bell, Witten and Fellows, 1998; Curzon, 2014a; Curzon *et al.*, 2014; Curzon and McOwan, 2017; Camp and Rader, no date) academic literature, and teacher input. It would be reviewed by Computational Thinking subject experts and Computer Science Leaving Certificate experts (content validity). Actions resulting from this review would contribute to the development of Version 2.

Post-primary teachers would evaluate Version 2 to ascertain its relevance and appropriateness for Transition Year students (content and consistency validity). Teacher feedback would result in the development of Version 3.

A workshop would next be conducted with teachers, to teach the course content and goals of Version 3. This version is then ‘tried out’ (see Figure 4-15) in schools by teachers to ensure the design's practicality and effectiveness. Feedback from this phase would result in the creation of Version 4 of the course

Version 4 would also be ‘tried out’ in schools and could be considered the summative version of the Computational Thinking course. It was planned that all the above components of the design would occur sequentially.

6.3 Recruitment

The above plan failed at the first hurdle, recruitment. The initial design (see Figure 6-2) required participants from three groups: (1) subject matter experts, (2) post-primary teachers and (3) post-primary students. This recruitment task was not undertaken lightly, a recruitment policy was developed, which considered many concerns from academic literature. Nieveen and Folmer (2013) highlight the following: 1) the clarification of participants' role, are they fulfilling the part of learner, critic or revisor, or playing more

than one role (Weston, McAlpine and Bordonaro, 1995). 2) The selected participants' ability to help answer the research questions. 3) The content experts location, i.e. belonging to a university different from mine (DCU) is recommended. 4) selective sampling (Nieveen and Folmer, 2013). This is advocated for design research as it ensures that feedback is as information-rich as possible (Nieveen and Folmer, 2013).

It was thus envisaged that the following participants would be required:

- Two Computational Thinking content experts (non DCU)
- Two Leaving Certificate Subject experts. These experts could also be content experts.
- Three post-primary teachers

The above number of participants was considered sufficient, as this phase's goal was to evaluate the intervention to correct deficiencies and shortcomings. For this reason, the number of participants was considered less critical: 'a remark of only one respondent could be highly valuable because of its salience. Small samples of respondents are usually sufficient if they are carefully selected' (Nieveen and Folmer, 2013, p. 163).

Regarding the types of schools and teachers selected, the following rationale was considered. It was envisaged that the chosen schools would be of different types and genders, i.e. mixed, single-sex, and DEIS (Delivering Equality of Opportunity in Schools) schools. It was planned to select teachers who had a range of experience with Computer Science and Computational Thinking. The above salient points were combined with advice from a colleague in DCU who had just finished implementing an 'App development outreach course' with post-primary teachers. She recommended the following regarding teacher recruitment

- participant teachers should 'self-select', rather than be nominated by their principals.

- teacher training workshops should occur close to the start of the course (her workshops had been conducted before the summer break, in the new term some of the teachers no longer worked at the participating school)

It was anticipated that by following the above criteria, the recruitment policy would help ensure the practicality of the learning material (Miles and Huberman, 1994; Nieveen and Folmer, 2013)

6.3.1 Recruitment Policy Implementation

I decided to recruit the teachers first. Students would be recruited from the teacher's classes/schools. Content Experts would be recruited next, preferably from outside of DCU.

6.3.2 Teacher Recruitment

Teacher recruitment started in September 2018 to run the course in January 2019. I had already one school, a DEIS school, that had contacted me (August 2018). They had heard of my study through word of mouth from a friend.

I decided to recruit the remaining teachers using the CESI mailing list, as I wanted the teachers to self-select onto the study. Members of the mailing list were asked on October 25th 2018 to complete a questionnaire to ascertain their understanding and attitude to Computational Thinking. They were also asked if they would be interested in piloting and collaborating on an introductory Computational Thinking course, (that assumes no previous knowledge). Eleven teachers responded affirmatively. Three were classed as ineligible: two were primary teachers, and one taught in the UK. The remaining eight teachers were subsequently emailed details about the study. Only three teachers responded to this follow-up email, all negatively. Of these three teachers: one teacher indicated she would be interested in participating in the summative phase. One teacher stated that the timeline did not suit, and the other, teacher X, took the time to ring me. He

was frank and asserted that I needed to change my design. Its current representation was not workable. When he responded to the CESI post, he was under the impression that I would teach the course. While he thought the course would be of great value to his students, he did not have the time to commit to both attending a workshop and teaching this course. He suspected other teachers would be of the same viewpoint.

Thus, after this first recruitment drive, I was faced with the possibility of having no participants, especially considering the DEIS school which I spoke to in September 2018, were now not responding to my emails. I took on board the advice and feedback from teacher X, and re-designed my approach, see Figure 6-3. (Note, whilst Version 1 was implemented with this research approach in mind, this was not the final overall design approach or student numbers see Chapters 7, 8, 9, and Figure 6-1).

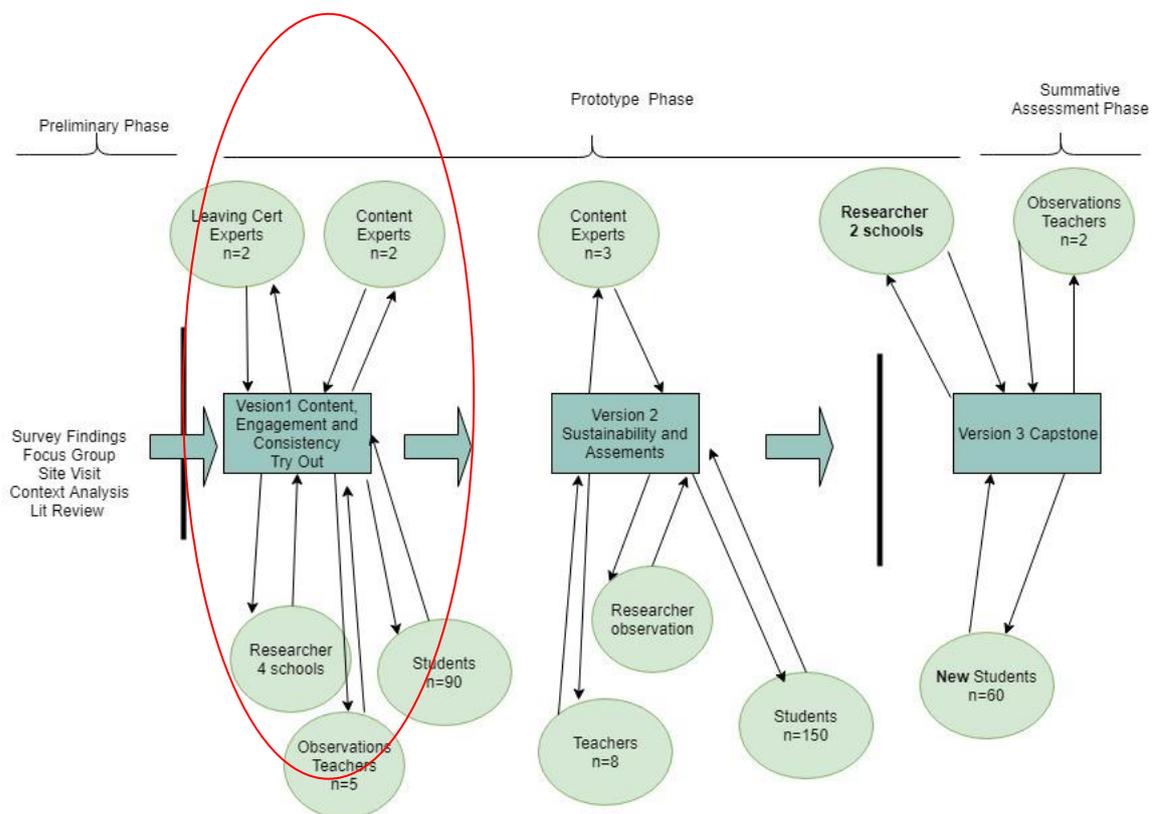


Figure 6-3 Re-design of Research Approach for my Design Research Study

I postponed the workshop idea and designed the course so that I, and not the participating teacher, would initially teach the course. The teacher’s role would now be one of observer and learner. The teacher would teach the course when piloting Version 3 (see Figure 6-3). I then re-sent emails to the eight eligible teachers who had originally responded to the survey indicating that an alternative design now existed for their consideration. Two of these teachers responded affirmatively.

Two more schools were subsequently recruited, one teacher sent an email after the initial recruitment drive asking to participate. The last school was recruited by convenience. A colleague had a child in this school, and when he heard about my difficulty with recruitment, he spoke about my research to the staff at the school. I now had the four schools I needed.

6.3.2.1 Teacher Profile

Table 6-1 illustrates the profile of the participant teachers in the schools. Three of the five teachers had an interest in teaching Leaving Certificate Computer Science or Junior Cycle Coding.

Table 6-1 Teacher Profile from participating schools

Teacher ID	School ID	Interest in teaching CS.	CS Qualification
1a	1	No	No
1b	1	No	No
2	2	Yes	Higher Diploma in Software Development
3	3	Yes	Computer Science degree
4	4	Yes	Higher Diploma in Data Analytics

6.3.2.2 Participating Schools

The following is a profile of the four participating schools. Table 6-2 provides a summary of the school types and the dates that the course ran in the school.

Table 6-2 Summary of school types and dates for participating schools

School Type	Dates: 2019	Lesson length	Student Numbers
School 1: All boy	25 th Feb, 4 th March, 11 th March, 25 th March, 8 th April	80 minutes	16
School 2: All boy	12 th March, 19 th March, 26 th March, 2 April,	120 minutes	28
School 3: All girl	8 th March, 15 th March, 22 March, 29 th March, 5 th April, 12 th April	80 minutes	30
School 4: Mixed	28 th Feb, 7 th March, 28 th March, 11 th April, 9 th May	40 minutes	25 (12 consent)

6.3.2.3 Schools

School 1 was selected using convenience sampling. Two teachers from the school were involved in this study, their Transition Year (TY) coordinator and a maths teacher interested in this area. Transition Year students self-selected on to the course. The TY coordinator confirmed that the class consisted of students interested in this discipline, and students who disliked PE and chose this course as an alternative in the timetable.

School 2 was an all-boys school. The teacher self-selected into the study and has an interest in teaching Leaving Certificate Computer Science in the future. Transition Year students self selected to participate in the course, and it ran during a free triple period the teacher had in his timetable. The students did not have a free period and thus had to catch up on the subjects they missed whilst attending the course.

School 3 was an all-girl school. The teacher self-selected into the study. This course was scheduled into the students' timetable. All thirty Transition Year Students participated in the course and also the study. The teacher was only available for the last forty minutes of each lesson, with different teachers supervising the first forty minutes session.

School 4 was a mixed-gender DEIS school. Students were timetabled for Computers during this period, and thus whilst all twenty-five students participated in the course, only

twelve consent forms were received. The participating teacher was not the Computer Science teacher, but she was interested in teaching the Junior Cycle Coding course and in setting up a Computer Club. The class was conducted in the participating teacher's science lab.

6.4 Data Analysis

As discussed in Chapter 5, the two outcomes for this study are the intervention and a set of guidelines (local instructional theory). The goal was to produce an intervention with certain characteristics (engaging, practical, effective, high quality, low threshold) and investigate what instructional theory and design helped meet these characteristics. Data gathered during the phase is related to the mediating processes and final outcomes highlighted in the conjecture map. The following questions guided my data collection and analysis to ensure it concerned the required phenomena (Bakker, 2018).

- Engagement: Are the lessons, i.e., content and unplugged activities engaging to students? (See 5.2)
- Practicality: Do the unplugged activities work in a school setting? They had initially been designed as part of an outreach program to interest students in Computer Science (Bell and Vahrenhold, 2018) Is the content usable and appropriate for the teachers? (see 5.1.2)
- Effectiveness: What are the teacher and student reactions? What did teachers and students learn? Did teachers use their new knowledge and skills (see 5.1.3)?
- Content: The validity of the content was assessed by expert appraisal and also by teachers. Sample questions included: Is the content appropriate for second-level students? Is the content relevant for teaching Computational Thinking? This criterion is a part of the quality criteria (see 5.1.1, 4.6).

The following instruments and methods were used to generate data in this prototype phase: interviews, teacher diaries, students' questionnaires, students' artefacts and researcher notes.

The task of documenting the evaluations and analysis performed on this data can best be treated under two headings: Revisions and Evaluation.

6.5 Revisions

During the prototype phase of Version 1, the revisions made to both the content and design of the course resulted from ‘On the Fly analysis’ and analysis that occurred during the Validation stage. These revisions and changes resulted in the creation of Version 2 of the course.

6.5.1.1 Revisions: ‘On the Fly Analysis’ (Stage 1)

This section is concerned with the analysis that occurred after each lesson (microcycle) twenty in total. This process shares parallels with lesson studies (Lewis, Perry and Murata, 2006). The main difference being the analysis in lesson studies leads to teachers' professional development, and with EDR, it leads to adding to scientific knowledge (Bakker, 2018).

The instruments used to collect this data were my field notes, informal discussions with teachers after lessons, teacher’s emails/journals and student engagement questionnaires. My researcher’s notes were verified at the teacher interviews. The outcome from this analysis resulted in revisions to the course content, design and instructional theory.

The course consisted of five units, which were piloted in four schools to eighty-six students, and five teachers. The following section contains five tables, one representing each unit. The tables provide context-rich narratives of the changes that occurred to the content and design of each unit based on its try out in the four schools (tables are read in landscape). The rationale for the changes are documented, and where appropriate, evidence is provided (using my field notes, teachers’ emails/journals). If changes were made based on discussions with teachers, these changes were validated at teacher interviews. For clarity, these validations are also included in the five tables.

Table 6-3 is concerned with Unit 1. A summary of the content of Unit 1 is as follows. It is concerned with generic problem solving and its subsequent relationship to Computational Thinking. The four pillars (components) of Computational Thinking are introduced, with three of them: decomposition, abstraction and pattern matching being discussed in detail.

Table 6-3 Content and Instructional changes to Unit 1

Notes on Course	Changes to content and design	Rationale	Validation: Researcher Notes Validated Teacher’s view on changes
<p>School 1 (all boy) 25th Feb 80 minutes</p>	<p>Too many slides, reduced from 31 to 14.</p> <p>Changed logic puzzles position: - Move the ‘marry’ puzzle and ‘bat and ball’ to start of lesson.</p> <p>Added pre-activation puzzles so students could learn by doing decomposition and abstraction. The activities added were the button sorting and the 30-second activity (see Appendix F).</p> <p>Added in placemats (see Appendix F) for collaboration and to involve all students in the reflection of activities.</p>	<p>Data: Researcher Notes ‘Class was too much teacher led. I was nervous and resorted back to what I knew, i.e., that of lecturing. Teacher 1a pointed this out.’</p> <p>‘My big criticism with the current design is that I need to pare it back. I have too many slides. I need more activities, and I need the class to be less directed by me.’</p> <p>‘Teacher 1a suggested that I use placemats to encourage collaboration and reflection.’ (see Appendix F).</p> <p>Design Ideas from Researcher Notes ‘New design idea. Start with an exercise or DEMO. Get them motivated straight away. More hands-on’</p> <p>‘Also noted that asking students for examples of decomposition, abstraction was met with blank faces. I needed to create “pre-activation” activities, that I could then refer back to so students could relate to them.’</p> <p>Data: Engagement Questionnaire:</p>	<p>Teacher 1a validated my notes during interview.</p> <p>Data: Teacher 1a Interview [ME] ‘So, for the purpose of the audio, I wrote comments about my first two class observations, and I have just got “Teacher 1a” to read them and to ask her was there anything there that she thought was inaccurate or anything there that she disagreed with.’</p> <p>[Teacher 1a] ‘Everything was perfect’.</p> <p>Pertinent points made by Teacher 1a in Interview about Lesson 1</p> <p>‘They didn’t even know what Computational Thinking was so, they were quite shy and slow to come forward. There wasn’t too many questions being asked of them. The fact that they actually couldn’t do anything for a long periods during the lesson they were passive, and they could tune out. I felt that very few of them could actually engage with you’ (Teacher 1a)</p> <p>Data Validation:</p> <p>‘Then when we mentioned the placemats for say, the second lesson, you were able to get a contribution from each of the kids and then nobody was isolated and whatever responses were given, were given as a group rather than as an individual, so no one was exposed or no one was vulnerable. The actual lesson, as we said,</p>

		The mean of the emotional engagement was the lowest score in this class for the whole course.	and you have noted yourself there, that there was too much information on the slides. Simple pictures or just a sentence or a puzzle, anything like that would evoke an awful lot more thought and you totally took that on board in the second lesson.’ (Teacher 1a)
<p>School 4 (DEIS School) 28th Feb and 7th March, 40-minute class</p> <p>The class on 28th Feb was Pre-tests, Class on 7th March did two unplugged activities, button and 30 seconds game as well as theory.</p>	Add recap after engagement activity each week.	<p>Data: Teacher 2</p> <p>‘a recap at the start to remind them of what they’re doing and what they did this week just to keep them focused’ (Teacher 2)</p>	<p>Data Email from Teacher 4 on 7th March 2019</p> <p>Validation of sorting buttons and 30-second pre-activation activity</p> <p>‘They were really engaged with the activities so definitely more of them. They seemed to really like them.’ (Teacher 4)</p> <p>Further email from Teacher 4, 3rd April 2019</p> <p>‘The students really enjoyed this, and they became a bit competitive, even the ones who were not so interested at the start. The specific questions were great to get them to think about how they solve problems especially playing '30 seconds' after they had done the riddles. Asking them how they guessed correctly was a great addition, they rarely get to think about that kind of thing.’ (Teacher 4)</p>
School 3 (all girl) 8 th March	<p>Fine-tune the Spelling Activity</p> <p>When giving engagement puzzles, ensure that I write up answers on board,</p>	<p>Data: Researcher Notes</p> <p>‘the first part worked really well. They were very much engaged’</p> <p>‘I am not happy with the spelling activity. This I believe needs to be fine-tuned.’</p>	Teacher not there for class, he was away. I had two different teachers supervising me.

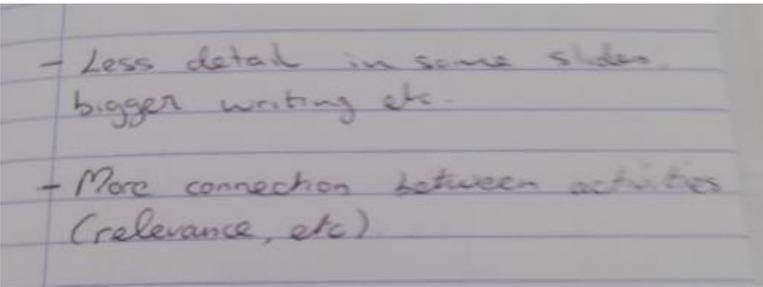
	<p>instead of just explaining them.</p> <p>Class-setup, students on benches-- placemats did not work as great here, as they made groups all on the one row.</p>	<p>‘I also in this class wrote up the answers for the puzzles, and I need to do this, as not all students get the answers. I need to demo more on the board. The class set up was not great. They were all on benches, so placemats were not the best way forward.’</p> <p>‘I think it works better when they see the link to CS’</p>	
<p>School 2 (all boy outside Dublin) 12th March and 19th March 120 minutes Unit 1 + some Unit 2</p>	<p>More fine-tuning needed with spelling activity.</p> <p>120-minute class (too long for this content)</p> <p>Need to link activities more to CS.</p>	<p>Data: Teacher 2 Email 12th March</p> <p>‘Students were also most engaged when the task at hand was very clear to them. The spell checker activity caused confusion for some but was introduced excellently with the spelling test. More scaffolding here to help students approach the problem with a strategy would help’ (Teacher 2)</p> <p>‘The length of the session is probably what caused some to lose interest at times. There was a lot of content for students to digest. I can see it being very effective broken into 40 minutes / 1 hour classes.’ (Teacher 2)</p> <p>‘The aspects explored such as decomposition/ abstraction etc. were well thought out. Perhaps at times students may have not understood their full context in a computational/coding sense and hence got slightly confused.’ (Teacher 2)</p> <p>Data: Researcher Notes:</p>	<p>Email 12th March Feedback from Teacher 2 which validated my changes of pre-activation.</p> <p>‘Overall I think students were very engaged and enjoyed the content. I believe it was set to the appropriate level. There was a great mix of theory and activity and there was always a clear relevance between both.’ (Teacher 2)</p> <p>‘The collaborative nature of the tasks worked very well. The videos used were interesting and concise. The PowerPoint was well formatted and the content was good.’(Teacher 2)</p> <p>Notes from Interview: Teacher 2</p> <p>‘That was incredible yeah. Sorting the buttons like and they see the difference in how they were going about size and shape you know like number of holes and stuff like that’. (Teacher 2)</p> <p>Data Researcher Notes 12th March:</p> <p>‘The setup worked great, magic trick, chocolate followed by theory, demo of kids and then they apply</p>

		12 th march 'I need to re-work the spelling assignment; they are unsure what they are supposed to do. I will need to stress the points of it. It needs to be scaffolded.'	the knowledge, Then we reflect and test it. I really love this model'. Data: Researcher Notes Week 2 part1: 19 th March (Lesson 1 to group that missed it the previous week.) 'I did out a template for spelling. This worked better, they really need to be scaffolded here, as they don't have a context at all. It is just too abstract for them.'
--	--	---	---

The original design contained the following elements: unplugged activities, activation, reflection (integration), application, demonstration and collaboration. The changes made to the design after the piloting of Unit 1 are as follows. The unit now starts with a puzzle. This has a dual purpose, 1) as a warm-up puzzle (Mingguang, 1999; Byrne, Kearney and Sullivan, 2018) and 2) a 'pre-activation' step. Pre-activation is the name I coined for memorable activities that establish a baseline of knowledge. This knowledge can then be activated later to build on new knowledge. Two other 'pre-activation' activities were added to this Unit 'button sorting' and 'thirty-second' game. These activities were necessary as I found that student did not have a common baseline that I could use to activate previous Computational Thinking knowledge. This finding correlates with Isu *et al.* (2017) who state that concepts central to Computational Thinking are alien to many K12 teachers. The use of placemats (see Appendix F) were also included in the unit to aid reflection (PDST, 2019).

Table 6-4 is concerned with Unit 2 which introduces the final core concept of Computational Thinking, algorithms. Logic, too forms a core part of this unit.

Table 6-4 Content and Instructional changes to Unit 2

Notes about Design	Changes to Content and Design	Reasons for change	Quality: Validation
<p>School 1 4th March (all boy) 80 minutes</p> <p>For this class, even though Unit 2. I introduced the button sorting, and thirty seconds activities, to ensure they understood the concepts of decomposition and abstraction (and to trial them on TY students, as I was worried, they might be too 'babyish'. This was first time they were trialed, see</p>	<p>Create a video to show spelling activity. (This will be a change to lesson1, but tried out in this lesson)</p> <p>Create a backup of all videos in case YouTube not available.</p> <p>Create screencasts of social media tweets.</p> <p>Less detail in slides, bigger writing</p> <p>Link more to CT</p> <p>Refine Logic puzzle to improve clarity, for example, with the logic exercise, let students know that they can refill the bottles with</p>	<p>Data: Researcher Notes</p> <p>Eclipse (a software tool used to run Java) would not run from my USB stick. I did not have permissions.</p> <p>Also no social media allowed, I wanted to show a twitter example.</p> <p>No sound from PC</p> <p>Feedback from Teacher 1b:</p> 	<p>Data: Researcher Notes on class</p> <p>‘The activities worked really well. They really engaged with the button and 30 seconds question’.</p> <p>New Content: ‘They seemed to like the card trick. The chocolate chilli went down very well. The big surprise was the emoji instruction, they really engaged with this. They also liked the water jug exercise. Both teachers were really positive. The idea of engage with activation, seems to be working very well.’</p> <p>Added two pre-activations.</p> <p>Discussion with teacher 1a and teacher 1b after class. Teacher 1a noted that as far as she could see, only one student not fully engaged, but the rest were very engaged. She noted how last week only 2 students engaged. She also noted how one student, who was a dark horse was the student who solved the water jug problem. She also noted how she liked how I brought the emoji example from last week back into play this week.</p> <p>Teacher1b: Interview about activities: validation of activities</p>

<p>microcycles Chapter 4). I had the time, as class timetabled for 3 periods, but only used 2 periods. So we went into period 3 for a little bit of this class</p>	<p>more water during the task.</p>		<p>‘So week two we had the guessing game, I just said, that was really engaging and fun and kind of got them into the lesson, so it was kind of a fun starter activity. Got everyone involved, which was really good. Sorting buttons was good, you gave no guidelines, they just went for that one ...’ (Teacher 1a)</p> <p>Interview: Card Trick (Teacher 1b)</p> <p>‘Yeah I don’t think me or them understood how that actually worked. I don’t think any of us knew where that was coming from but I think it was interesting for them to – it kind of blew their mind a little bit yeah and they blindly followed yeah exactly. Just followed instructions isn't it’(Teacher 1b)</p>
<p>School 3 15th March (All girl School, 80 minutes). (Re-did spelling activity from week 1, to link it better to CS)</p>	<p>No changes to content, from this class</p>		<p>Data: Researcher Notes: Validation of unplugged activities</p> <p>‘The magic trick and jar game seemed to go down really really well. They were very much invested in it. They were really excited when it worked.</p> <p>I then got then to write the emoji code again. I only saw one student not participating in this activity. They all asked to participate in me demoing their instructions. They were all totally invested in this. This worked really really well.’</p> <p>I had a quick word with Teacher and he felt the lesson went really well</p>

<p>School 4 28th March (40 minutes)</p> <p>Chilli-chocolate, card, emoji</p>	<p>No changes made to content or design</p> <p>Researcher Reflection ‘The learning here was with my teaching.</p> <p>Some of the children in the school do not handle criticism well. I had to change how I gave them feedback on their emoji pictures.’</p>		<p>Design Validation: Peanut video essential here as they need to see success</p> <p>Teacher Data: Feedback from Teacher 4 by email: 3rd April 2019 ‘Session 3 - The students are more settled into the idea of the classes now. They enjoyed the competitive nature of this class too and I was surprised how much effort some of the groups put into the in part of the lessons where they had to give instructions. Introducing the video of the dad with his kids first was a great idea to explain the logic of it to them.’</p> <p>Researcher Reflection: Peanut video essential here as they need to see success</p> <p>‘They all wanted feedback. I realised showing them what success is important, and I feel this is vital for this class, and also the design of the course. Letting them explore this themselves would be disastrous.’</p>
<p>School 2 19th March (4 hrs today in school as repeated lesson 2 for some students, before doing</p>	<p>No changes</p>		<p>Librarian was my chaperone for repeat class. Data: Researcher Field Notes My (researcher) observations.</p> <p>I did out a template for spelling. This worked better, students really need to be scaffolded here, as they don’t have a context at all. It is just too abstract for them.</p>

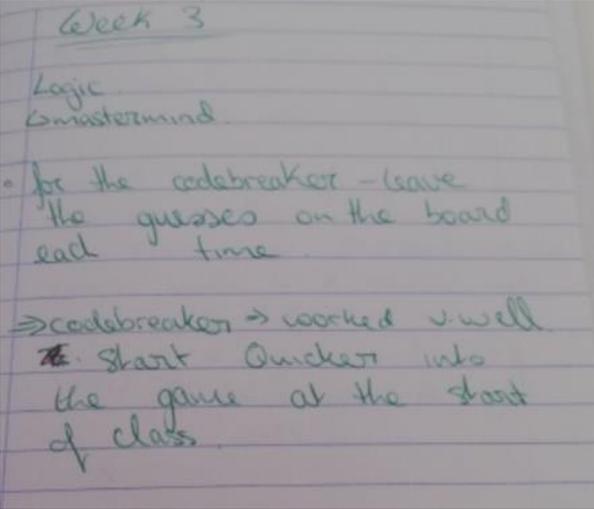
lesson 3 with them all)			The algorithm part was once again very successful. The logic part is also a big success. One of the students took the card trick and tested if it worked for 3 cards, and then 5 cards. This was very impressive. They also came up with some great algorithms.
-------------------------	--	--	---

There was no real changes to the design from Unit 1, although Unit 2 ‘try out’ served to validate the new model of warm-up puzzles with pre-activation functionality. Clarity was a feature that stood out as being very important. Recaps at the start of a lesson was also a feature added.

Table 6-5 is concerned with Unit 3. Its aims are to introduce students to some standard algorithms that are considered as fundamental algorithms in Computer Science. They are binary and linear search. This unit also allows students to see Computational Thinking in action with a group activity titled ‘Searching to Speak’.

Table 6-5 Content and Instructional changes to Unit 3

Notes about Design	Going forward: changes to content	Reasons for change	Quality: Validation
School 1 11 th March (all-boy school) 80 minutes	Need to link the unplugged activities to Computer Science	Data: Teacher 1b	Validation on codebreaker: Teacher 1b interview: ‘The code-breaker worked really well.’ ‘I thought that was a good week’ Data: Researcher Notes

<p>Note Codebreaker was done here in week3, but in other schools, this was done in week 2.</p>	<p>Change ‘Searching to Speak’ activity: too abstract</p> <p>Design concept: Activity then theory.</p> <p>Have a warm up exercise every class. After recap, exercise straight away.</p> <p>Link examples with Computer Science again, just to highlight the relevancy. I am giving them puzzles but I need to link them to CS and (CT more)</p> <p>Teaching Hint:</p> <p>Teacher 1b suggested that with the Codebreaker demo, I show the computer guesses on the board, as it helps students to see a pattern.</p>	 <p>Data: Researcher Notes</p> <p>‘The Searching to Speak activity, I believe, was a little too abstract, they did not seem as engaged. But the teacher felt it worked really well.’</p>	<p>‘The Codebreaker exercise worked really well. They saw how they were making educated guesses. I did tell them that the computer had a strategy, and they did try to find this out’</p>
--	--	---	---

<p>School 2, 19th March, (all boy school, 120 minutes)</p> <p>Mix of lesson 3 and lesson 4.</p>	<p>I still need more work on the Searching to Speak activity.</p>	<p>Data: Teacher 2: Email 20th March Feedback from teacher regarding the Searching to Speak Activity (also known as locked-in syndrome task)</p> <p>‘The locked in syndrome task was also very good but perhaps went on too long for some. The lads tackled the problem, but I think they knew there was always going to be a more efficient way of solving it and hence lost some interest after a few attempts. The learning involved was excellent and its very appropriate to the content.’ (Teacher 2)</p>	<p>Validation for video (die hard and google duplex) and linkage of activities.</p> <p>Data: Teacher 2: Email 20th March ‘The opening task with the water jug logic puzzle was very effective. The lads worked very hard in groups. They enjoyed the video of Die Hard. I think it helped them relate to the task more.’ (Teacher 2)</p> <p>‘The code breaker game also went down very well. As mentioned last week the competitive aspect really engages the lads. I thought it was very valuable to mention beforehand that the students will vastly improve their logic and problem- solving abilities. I think there was great learning involved in the context of binary searching and inductive and deductive reasoning.’ (Teacher 2)</p> <p>‘There was a clear connection to the first session by reinforcing the use of the computer science terms such as algorithms, pattern matching and abstraction etc. This definitely helps reinforce the lads understanding of the terms.’ (Teacher 2)</p> <p>‘The lads loved the Google Duplex video and the topic of AI. They love futuristic concepts and what new ways computer science is changing technology. Overall a brilliant session, very active and engaging.’ (Teacher 2)</p>
<p>School 3 22nd March</p>	<p>Add in a video before ‘Searching to Speak’ demo,</p>	<p>Data: Researcher Notes</p>	<p>Data: Researcher Notes</p>

<p>(all-girl school)</p> <p>Searching to Speak</p>	<p>as a lead-in/ hook, to activity</p> <p>Create a fact sheet for teachers about Dominique, as students wanted more information</p>	<p>‘The seemed really engaged with searching to speak, asked loads of questions’</p>	<p>They seemed really engaged with searching to speak, asked loads of questions. One girl also asked for more time to do the codebreaker task. They really wanted to try it out. In conversation with Teacher 3, he mentioned that now with the new junior cert and classroom-based assessments, that the computer room is being used more, as they need to use powerpoint. I believe I will keep the template for searching to speak from this class.</p>			
<p>School 4 11th April (DEIS Mixed)</p>	<p>Changed Codebreaker worksheet to make filling out table easier. Add in the following graphic</p> <table border="1" data-bbox="434 683 757 906"> <thead> <tr> <th data-bbox="434 683 757 724">CodeMaker Response</th> </tr> </thead> <tbody> <tr> <td data-bbox="434 724 757 766">✓ Colour & ✓ Loc</td> </tr> <tr> <td data-bbox="434 766 757 906">× Loc & ✓ Colour</td> </tr> </tbody> </table>	CodeMaker Response	✓ Colour & ✓ Loc	× Loc & ✓ Colour	<p>Data: Researcher Notes</p> <p>Two students got confused on data to give to code breaker</p> <p>Engagement scores: all mode values 4, all median values except for one item were also 4</p>	<p>Data: Interview Teacher 4: Validation of design i.e. demo/example first before activity concept</p> <p>‘I think they loved it especially if you did an example first or if you showed them something and they had to figure out how it was done, either the code breaking thing or any of the games and activities, they were engaged. They did want to take part and especially when, as you said already when they understood what the outcome should be, they were all engaged. Because they were working in groups, they were kind of competing against each other which helped bring them in. Anyone who wasn’t engaged it helped bring them in. They wanted their group to win. So, everyone did get involved’ (Teacher 4)</p> <p>Data: Researcher</p> <p>‘I had the attention of about $\frac{3}{4}$ of the class. At first, they were reluctant to participate, did not want to write out the logic answer for the bottle puzzle, but in the end, we were able to encourage them by setting it up as a competition with an award of smarties.’</p>
CodeMaker Response						
✓ Colour & ✓ Loc						
× Loc & ✓ Colour						

The importance of linking the warm-puzzles to Computational Thinking was highlighted. The use of videos as a lead/in hook to activities was used successfully in Unit 2, and was also piloted into Unit 3 to give an overview of Jean-Dominique Bauby background. Once again, this strategy was a success. The approach of providing demos of activities and concepts before application also first proved really successful, especially in the DEIS school. The Codebreaker game was first demoed online before the students did the game on paper. A similar approach was used in week 2 where the students saw a video on ‘exact instructions’ before they wrote their own algorithm. Teacher 4 highlighted the success of this strategy specifically in connection with clarity. The instructional design now has the following setup: unplugged Activities and clarity (Transparency) are immersed in a cycle of **P**re-activation, **R**eflection and **T**heory, **D**emonstration, **A**pplication and **R**eflection.

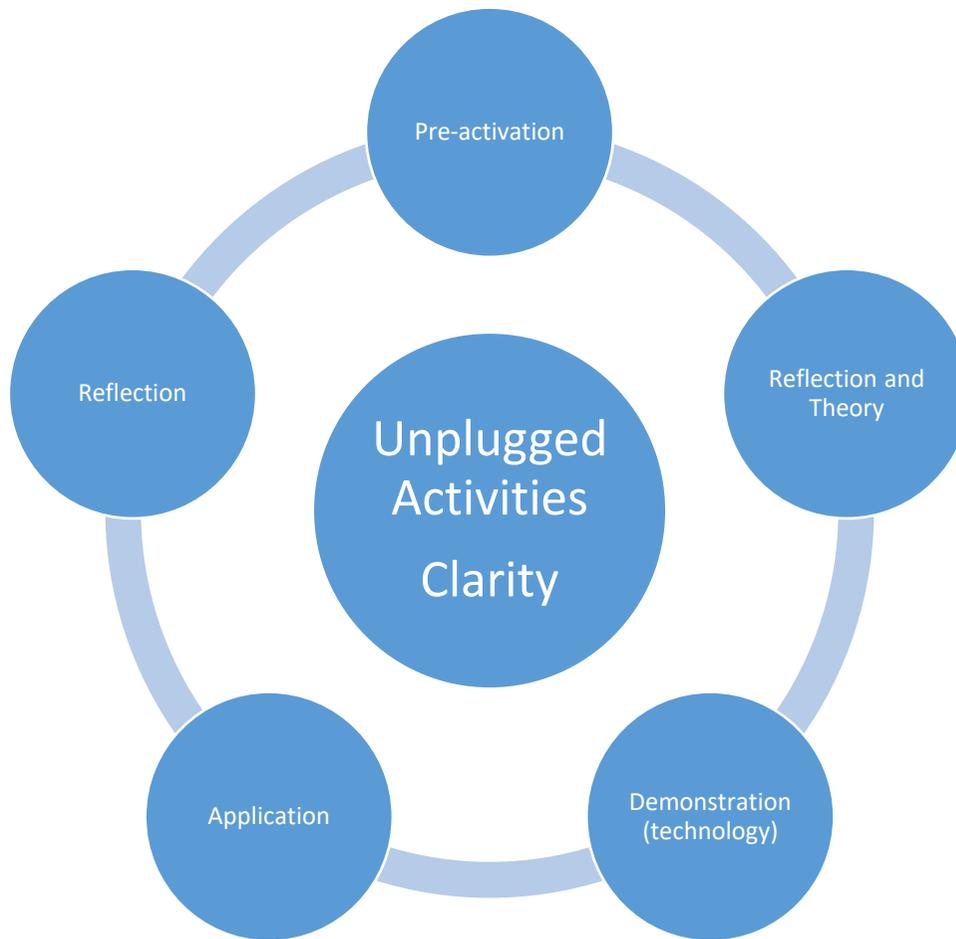
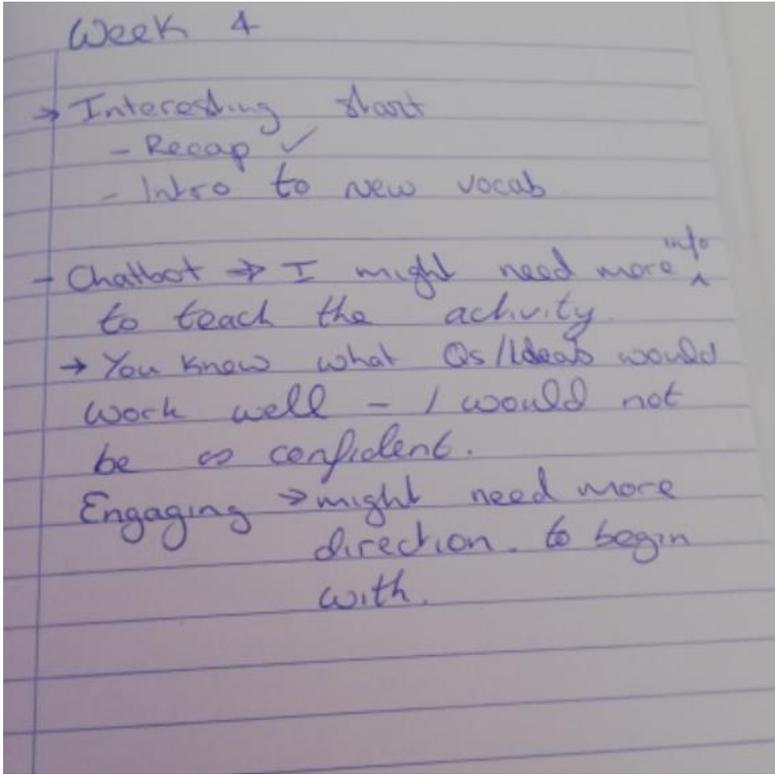


Figure 6-4 The emerging instructional framework

Table 6-6 is concerned with Unit 4. This unit aims to introduce students to artificial intelligence and ethics. The goal here is to discuss if the thinking used to create algorithms should be moral as well as valid (logical).

Table 6-6 Content and Instructional changes to Unit 4

Notes about Design	Going forward: changes to content	Reasons for change	Quality: Validation
<p>School 1 25th March 2019</p> <p>AI: ChatBot Moral Machine</p>	<p>Teachers will need more explanation in teacher guide, on how to teach the chatbot activity.</p> <p>Also need to give more examples to students, so they know how to complete the chatbot activity</p>	<p>Data Teacher 1b</p>  <p>Engagement score lower than normal:</p>	<p>Data: Researcher Reflection:</p> <p>Engagement score not as high as normal. This may be due to the chatbot activity, as it needs more clarity</p> <p>Data: Researcher Notes: Validation of AI content</p> <p>‘They all loved seeing the new google AI duplex. They all loved the driverless car and how their morals changed, i.e. from trolley problems to when it came to them buying a car.’</p>

		<p style="text-align: center;">Descriptive Statistics</p> <table border="1"> <thead> <tr> <th></th> <th>N</th> <th>Minimum</th> <th>Maximum</th> <th>Mean</th> <th>Std. Deviation</th> </tr> </thead> <tbody> <tr> <td>meanEmo</td> <td>13</td> <td>2.33</td> <td>5.00</td> <td>3.8205</td> <td>.67516</td> </tr> <tr> <td>meanCog</td> <td>13</td> <td>3.00</td> <td>5.00</td> <td>3.9615</td> <td>.55758</td> </tr> <tr> <td>meanBeh</td> <td>13</td> <td>3.00</td> <td>4.67</td> <td>3.8205</td> <td>.48334</td> </tr> <tr> <td>meanEngage</td> <td>13</td> <td>3.38</td> <td>4.63</td> <td>3.8558</td> <td>.41723</td> </tr> <tr> <td>Valid N (listwise)</td> <td>13</td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>		N	Minimum	Maximum	Mean	Std. Deviation	meanEmo	13	2.33	5.00	3.8205	.67516	meanCog	13	3.00	5.00	3.9615	.55758	meanBeh	13	3.00	4.67	3.8205	.48334	meanEngage	13	3.38	4.63	3.8558	.41723	Valid N (listwise)	13					
	N	Minimum	Maximum	Mean	Std. Deviation																																		
meanEmo	13	2.33	5.00	3.8205	.67516																																		
meanCog	13	3.00	5.00	3.9615	.55758																																		
meanBeh	13	3.00	4.67	3.8205	.48334																																		
meanEngage	13	3.38	4.63	3.8558	.41723																																		
Valid N (listwise)	13																																						
<p>School 2 26th March (all-boy)</p> <p>120 minute class</p> <p>Unit 4 and Unit 5</p>	<p>Need to make chatbot exercise clearer. Some groups, could not decide on what topics to pick, need to give more examples.</p> <p>Need to highlight relevance of ethics to CT as some students questioned this.</p> <p>For exercises, students were reluctant to write down their views.</p>	<p>Data: Researcher Notes</p> <p>‘Students questioned the relevance to CT, so I believe I need to stress more why I am asking these questions.’</p> <p>‘Some students were very reluctant to write down their views, (they pretended they did not have pens, even put them in their pockets so that they did not write) so I probably need to rethink how I assess this’</p> <p>‘They seemed to be very engaged with chatbot, showing the lady Macbeth code I believed really helped with their understanding.’</p> <p>‘Some other random thought, when I ask them to write down the instructions to write a cup of tea I heard a student complaining. Other students mentioned that they had done this before in HTML class. ‘</p> <p>Some had great difficulty deciding on what to do their chatbot on.</p>	<p>Data: Teacher 2 Email 29th March</p> <p>‘The opening with the chatbot and using the Q-Cards was good for getting the lads engaged. Once they were clear about the task at hand they really got into it.’ (Teacher 2)</p> <p>‘The theme of Ethics was really interesting and thought provoking for the lads. The worksheets on driver-less cars were interesting and I could tell they enjoyed the moral dilemmas presented to them. I thought it was excellent that you mentioned the book 'How to be Human in the Age of Machines'. This allows the lads to explore the area further. I also thought it was good how you were able to tackle the question of 'When would this ever happen?' by providing real life stories. Bringing aspects like Stanislav Petrov into the discussion was great and also the use algorithms in crime etc.’ (Teacher 2)</p>																																				

	Remove the exercise, writing down instructions for cup of tea, explain instead. (Lesson 5)		‘The session relied a good bit on discussion and I think that was appropriate. Overall I thought the content was good and it was set at an appropriate level. I still believe that it is a lot for students to consume in 2 hours and as such it would suit 40 minute to 1 hour class periods very well.’ (Teacher 2)
School 3 29 th March (all-girl)	No changes	Changed chatbot format to highlight how important it is that the ‘machine learns’ The Engagement score for this class was quite high relative to other classes. The girls loved the creative aspect of the ChatBot.	Data: Researcher Notes ‘They seemed to like the chatbot. My take on this approach was to highlight AI, how superior it is, and how it is important that the machine learns. Unlike school 2, they had no problem deciding on topics. I saw chatbots from vampire diaries to movie stars. One group found their chatbot successful, the others saw that it had limitations. The AI phone call was once again very effective, The 20 questions also went down very well. There are about 4 girls who have no interest, but the rest seem very engaged. They were all engaged in the questions on ethics, and were very thoughtful and rational in their responses.’
School 4 9 th May (DEIS, mixed)	Class had different format, no activities. Highlighted the	Data: Researcher Notes: ‘Strange class today, I had not planned to do any teaching as I thought it was last class. This is a strange class in that half do not want to be in school but the others seemed to listen to me and are engaged in what I say. They were very noisy but their responses were appropriate and relevant. I would have liked to do the chatbot but planned class differently as I thought it was the last class. We did artificial intelligence, driverless cars’ Very discussion based. The class teaching did not reflect the current design	Data: Teacher 4 interview: Validation for Content and Course ‘I don’t think if you said a machine is really smart..... that was one of the things that I really thought came across every single week; that machines had to be programmed and they are only as smart as the people who programme them even though they are learning. So, I think that is something that they should have taken away. They seemed really interested in the ethics thing.... I think

		<p>The engagement scores reflect that I had not planned class, and that there was no unplugged activities.</p> <p>The majority of students indicated they were not bored with the class, and were focused on the learning. However the majority of students did indicate the class was not exciting, and they talked to others about content not related to the course.</p> <p>On a positive note, this class highlighted the need for the structure and unplugged activities</p>	<p>that might be something that might stick with them a little bit. They were all really engaged with the driverless cars. I think if I was to say, what is Computational Thinking? They might say, 'I have never heard that word before'. You know, you would have to kind of pull it out of him a little bit before, but I would say they would definitely know that now. They definitely learned that.'(Teacher 4)</p>
--	--	---	---

This unit did not start with a puzzle, but with a chatbot video that served the same function, it was used as a warm up and also as a pre-activation steps. Students now had a baseline of knowledge on chats bots. Once again, the approach of using videos as hook for activities was successful.

Table 6-7 concerns Unit 5. The purpose of this unit is to introduce students to the foundations of programming, specifically pseudocode. The goal of this lesson is to link Computational Thinking and programming.

Table 6-7 Content and Instructional changes to Unit 5

Notes about Design	Going forward: changes to content	Reasons for change	Quality: Validation
<p>School 2 2nd April All boy outside Dublin</p>	<p>Need more content in cheat sheet, as students needed more examples and options to use when writing their code.</p>	<p>Data :Teacher 2 Email:</p> <p>‘Given the complexity of the content covered I think the lads learnt a lot! This is based off how they approached the task of writing pseudo code for Caesars code. Some naturally found it easier and as such I feel they got more from the activity. Others who lacked interest required more attention and guidance. This is a common occurrence within any subject/topic.’ (Teacher 2)</p> <p>Data: Researcher Notes:</p> <p>‘The programs I got back were very insightful, especially when you consider that a lot of concepts were introduced in a short period of time. I will check the pre-assessments to find out how many have programmed before. They really tried their best in this task. Some take-homes for this task is that I need to add more content into cheat sheet’</p>	<p>Data: Teacher 2 Email: 3rd April Validation of Content</p> <p>‘I really enjoyed today's session. I also think the lads really enjoyed it. They were engaged from the outset.</p> <p>Recapping on what has been done so far was worthwhile. It gave the guys a good reminder of all the content they have covered.</p> <p>I think the approach to programming used was excellent. The lads were so engaged when you told them that you were going to teach them the basic fundamentals of any programming language.</p> <p>The structure of the content and how it was presented was very suitable.</p> <p>You explained the 5 fundamentals such as input, output etc and then built on these consistently ensuring that the lads were clear in their understanding of each. Breaking down each aspect made it easy to follow.</p> <p>You explained pseudo code incredibly well and the 'cup of tea' instructions made it clear what you meant.</p>

			<p>The cheat sheets were also very valuable, it gave a very advanced feeling to today's class and gave the lads exposure to coding terminology. Given the complexity of the content covered I think the lads learnt a lot! This is based off how they approached the task of writing pseudo code for Caesars code.</p> <p>Some naturally found it easier and as such I feel they got more from the activity. Others who lacked interest required more attention and guidance. This is a common occurrence within any subject/topic.</p> <p>The majority of the students in attendance did not have any/much coding experience before this 4 week course, yet they were able to format basic pseudo code and identify bugs or errors. I think this indicates massive learnings in a number of contexts such as logistical problem solving skills and strategic planning abilities. The lads were using computational Thinking to address the task.</p> <p>From a teachers perspective, I can see the advantages of approaching programming through teaching students these skills in an unplugged fashion. It is certainly something I will be adopting in my practices. The students will benefit from it and it will make the teachers job easier as progression occurs while teaching coding.' (Teacher 2)</p>
--	--	--	--

<p>School 3 All-girl school Date 5th April</p>	<p>Need more scaffolding of max problem. Put more answers on cheat sheet</p>	<p>Researcher Notes: 5/4/19 With reference to the max problem. 2 girls from 2 separate groups volunteered to show me their solutions</p>	
<p>School 1 8th April (all boy)</p>	<p>Pseudocode exercises still need more scaffolding. Cheat sheet needs to be re-designed. Going to write the blocks of code they need and then they pick the order (similar to scratch(</p>	<p>Teacher who is new to the subject found this difficult. Researcher Notes; Advice from teacher, is that we need to scaffold this exercise more, as students new to programming needed a lot of guidance. One of the students mentioned to me that this was the hardest class of the course. Teacher 1b also mentioned to me that she would need a lot of scaffolding too for this class. When I was speaking to students they all seemed very engaged, so that was very positive. They also all did the max question by themselves</p>	<p>Data Feedback from Teacher 1b</p>

WK 5

~~Notes~~

Pseudo code

- input
- output
- Sequence
- repetition
- condition

o how to find the max no.
(maybe change to min (right days))

↳ Student activity - Difficult.
possible to break down more?

→ Needs more explanation

→ I think boys were unsure
how to write it down.

Maybe a guided sheet would work
eg Variables: —
—

This way it would
prompt them with how
to structure it.

Cheat sheet → too complex
the boys found it
to follow.

Maybe needs a clear laid out
Example.

(would find this lesson
hard to teach)

The cognitive load of unit 5 (similarly to programming) proved too difficult for many students (Alexandron *et al.*, 2014). This unit was subsequently revised to investigate (in Version 2) if updating the cheat sheets to mirror blocks of code similar to what is found in the Scratch Integrated Development Environment (IDE) would make the activities easier for students. The activity, writing an algorithm for making a cup of tea, will be changed, as students did not find it interesting. A summary of all ‘on the fly’ revisions is found in Appendix G.

6.5.1.1.1 Student Engagement Questionnaires

Engagement is a characteristic that was important to this course from the onset. The course content and instructional design were designed with engagement in mind, and the course was evaluated to see if this objective had been achieved. During the piloting of Version 1, students were issued with an engagement questionnaire after each lesson (see 4.5.2.1). This engagement questionnaire served three purposes 1) to provide the student voice during the piloting of each lesson, 2) to provide feedback on the course content and design 3) to be triangulated with data from engagement captured from teachers and qualitative data from students.

The mode and median values for each of the eight items per school are tabulated. A snippet of the results from school 3 are shown below, with full results from each school displayed in Appendix H.

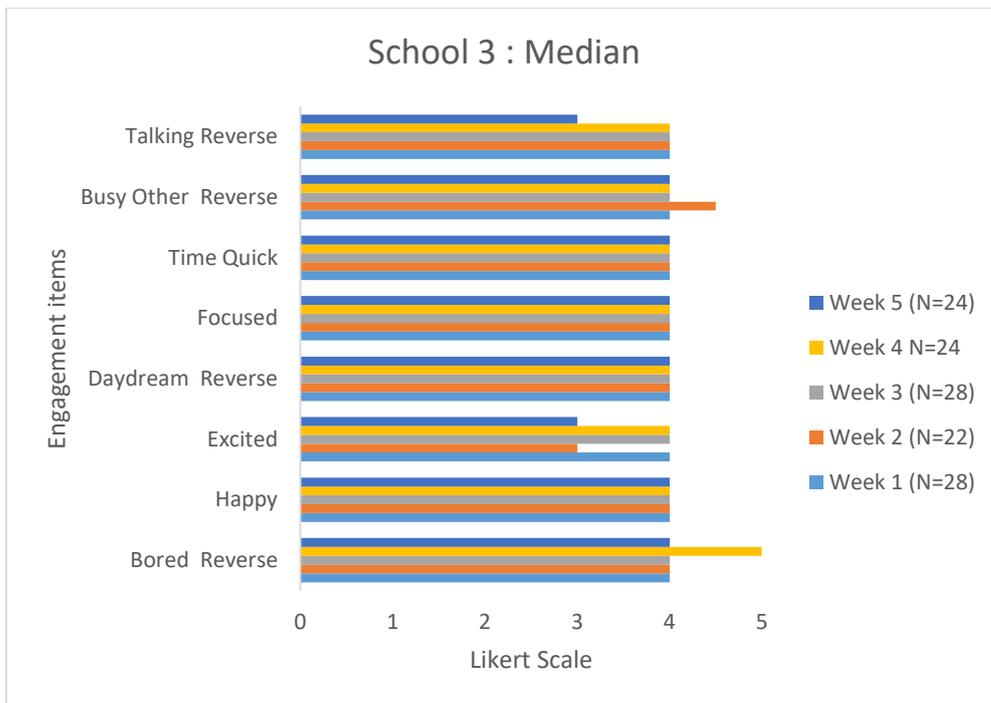


Figure 6-5 School 3 Median values

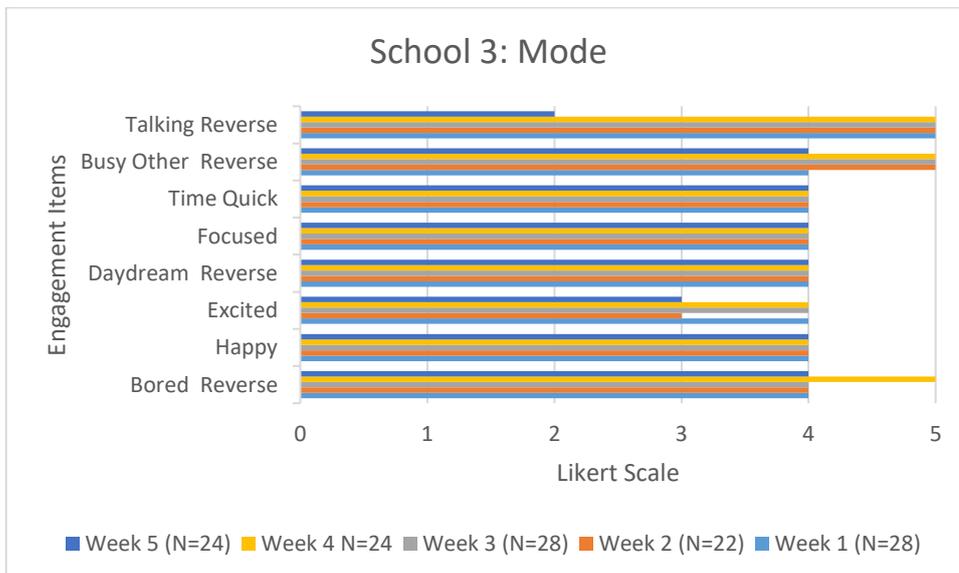


Figure 6-6 School 3 Mode Engagement values

Note: Week 1 Busy Others, Week 3 Focused, Week 5 Talking are multiple modes, smallest value shown

What stands out in the results from each school (see Appendix H) is that for the majority of cases, the mode and median value for each item is either 4 or 5. This indicates that students on the whole, found the content interesting, were focused on the activities, and felt the time

went quickly. The weeks where some items returned a mode value of less than 4, correspond with specific feedback for improvement from both teachers and students. Of note, Week 1 in School 1, was reported by the observing teacher as lacking in engagement. This lack of engagement is indicated with the 'Excited' and 'Time Quick' items having a mode and median value of 3.

They didn't even know what Computational Thinking was so, they were quite shy and slow to come forward. There wasn't too many questions being asked of them. The fact that they actually couldn't do anything for a long periods during the lesson they were passive, and they could tune out. I felt that very few of them could actually engage with you (Teacher 1a).

Week 5 was reported by one teacher and many students in School 3 as being too difficult, see section 6.5.1.1 and section 6.6.2. This feedback corresponds with the 'Excited' and 'Busy talking to others about non-related things' items having a median and mode value of 3 and 2 (multi-mode, smallest value shown). The engagement score for week 4 in School 1 was also lower than normal, the fact that students had difficulty coming up with ChatBot ideas may be the reason here. The opposite was true for week 4 in School 3 (the all-girl school). The students embraced the creative element and had great fun creating chatbots based on tv shows such as 'Vampires Diaries' and actors.

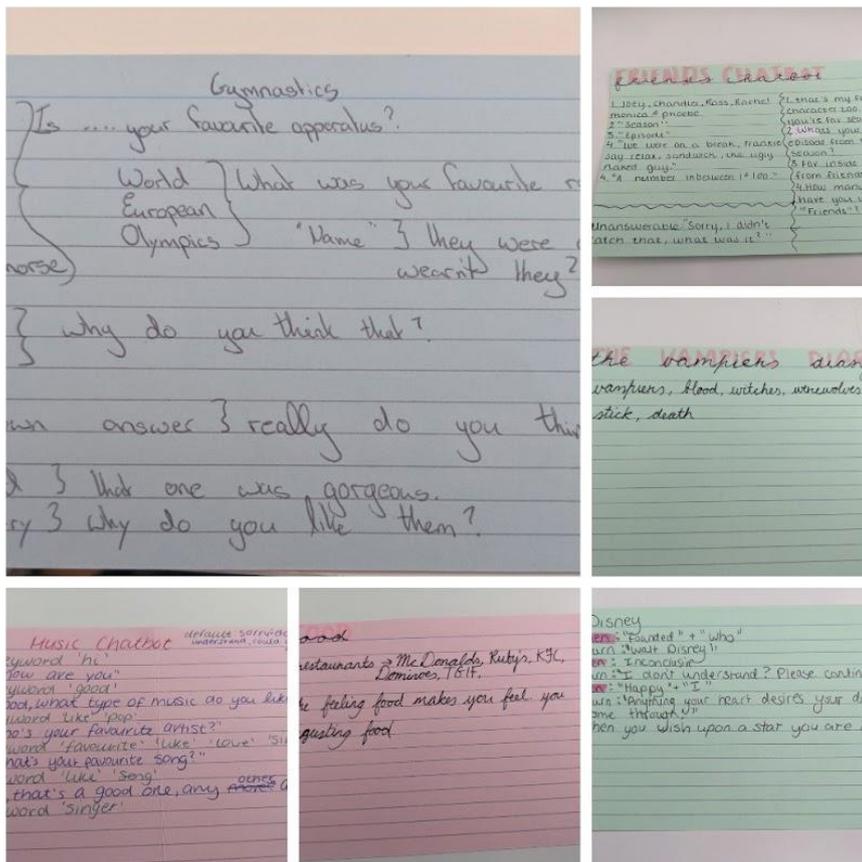


Figure 6-7 A collage showing the creation of scripted chatbots

The engagement scores reflect this. It was the only week in this school where three items had a mode of 5. The engagement scores for the last week in School 4 reflect the unstructured class where it was very discussion-based, and there was no application of knowledge.

6.5.1.1.2 Summary

The engagement questionnaire served to provide the students' perspective during the course. Their results support the view that students found the content interesting and engaging.

6.5.1.2 Revisions: Validation Analysis (Stage 2)

Revisions to the content and design were also performed as a result of 'validation analysis' (see Stage 2, section 4.7.2). To aid clarity and to avoid repetition, these revisions are discussed under different headings (see section 6.5.1.3. Quality and section 6.5.1.4 Content

Revisions). The reason for this was the revisions performed in Stage 2 were validated in the Stage 3 analysis.

6.5.1.3 Quality

An aim of the Computational Thinking course is that it is of high quality. Version 1 of the course was evaluated for ‘content validity’ (Nieveen, 1999b). Two content experts were recruited to review the teacher guide for Version 1 of the course. They were carefully selected due to their Computer Science (CS) expertise and their experience with the CS Leaving Certificate course or Computational Thinking. Nieveen and Folmer (2013) highlight how this selection is important as ‘a remark of only one respondent could be highly valuable because of its salience’ (2013, p. 163). The experts were given a copy of Version 1 of the teacher guide prior to an interview. A summary of this guide is documented in section 5.7.

The experts came from different Irish Universities. They answered questions related to the relevancy of the course topics to Computational Thinking and its potential for developing students’ understanding of Computational Thinking. Their generated data was analysed in two stages, 1) the ‘Validation stage’ using a deductive content analysis approach and the 2) ‘Standard stage’ using thematic analysis (see Chapter 5).

6.5.1.3.1 Participants

Contextual information related to the experts is discussed in this section. Although comments attributed to them are cited without a name, both individuals permitted their names to be used.

At the time of the interview, Expert 1 was an Assistant Professor at Trinity College Dublin with responsibility for their MSc Technology and Learning course. He has experience in teaching Computer Studies in post-primary schools and has a specific interest and research expertise in Computational Thinking. He was involved in research into Computational

Thinking for Life. He was interviewed in person on the 8th May 2019. The interview took about two hours.

Expert 2 is an Assistant Professor in Computer Science at University College Dublin. He has extensive experience in teaching Computer Science at third level. His main research interest is Computer Science Education, and he has recently co-authored a textbook for the Irish Leaving Certificate Computer Science course. He was interviewed on 19th June 2019. This was conducted on the phone and took about one hour.

6.5.1.3.2 Semi-Structured Interviews

The data collected from the content experts was qualitative, i.e. semi-structured interviews. The content questions were influenced by questions from a design-based research study conducted by (Fauzan, Plomp and Gravemeijer, 2013), with the quality questions added by me. The questions are as following:

Quality Question

1. Can you confirm that you've read the document and how fresh it is in your mind?
2. Can you allude to your knowledge of Computational Thinking?

Questions regarding the content of the course

3. Does the content of the course include relevant subjects and topics?
4. Does the content of the course reflect the key principles of Computational Thinking?
5. Has the course the potential for developing students' understanding of Computational Thinking?

6.5.1.3.3 Standard Analysis: Thematic Analysis Findings

Braun and Clarke's (2006) six-step thematic analysis framework was followed using the software tool NVivo.

- Step One: Become acquainted with data
 - I re-read the two transcripts to become familiar with the data.
- Step Two: Generate initial codes
 - I created three of the original categories (from Validation stage) straight away, ‘Corrective Feedback’, ‘Quality’ and ‘Validations’. I then coded the interviews into these categories. This was done to ensure I had not missed any revisions during the ‘Validation Stage’ (see Appendix I). To aid with this verification, I ran a cross-code analysis to identify the similarities between the two experts. A snippet of this cross-code analysis is shown in Table 6-8.

Table 6-8 Cross Code Analysis table for Expert 1 and 2

	A: 190619_001_Expert_2	B : 190508_001_Expert_1
1 : Abstraction	0	8
2 : Academics	0	3
3 : activity	0	3
4: Algorithms	0	3
5: Assessment	0	3
6: Computer and CT	0	1
7: Corrective Feedback	11	29
8: Layout	0	2
9 : Question	0	3
10: Course Subjects	3	4

- I then started afresh, where I worked inductively to generate initial codes. Before I completed this task, I assessed my strategy and revised my coding technique.
- Step Three: Search for themes
 - The next step was concerned with capturing themes, patterns that illustrated a significance about the content quality. Using NVivo, I created categories related to the codes (from step 2) and reorganised the NVivo nodes (see Chapter 5 for coding style). Five initial categories were created: ‘Subjects,’ ‘Course Design’, ‘Language’, ‘Pedagogy’, and ‘Clarity’.

Phase 2- Developing Categories

- ◆ Name ▲
- + ○ Category Content Topics Subjects
- + ○ Category Course Design
- + ○ Category Language
- + ○ Category Pedagogy (Teaching)
- + ○ Clarity

Figure 6-8 Initial themes created for the content experts analysis (NVivo 1.4.1)

- Step 4: Review Themes
 - The themes were next reviewed to ascertain the following: Are they appropriate? Do they support the codes? Is there overlap? Are there sub-themes? (Maguire and Delahunt, 2017). This review resulted in the ‘Language’ category/theme being re-categorised as a sub-category under the main ‘Clarity’ theme, as it already had a subtheme of ‘Clarity’.

Name	Files	References
Category Content Topics Subjects		0
Category Course Design		0
Category Language		0
Advice		1
Clarity		1
Abstraction		1
Clarity-Information		2
Definitions		2
Confusion		2
Language		2
Need to define CT		2
Sorting		1
Category Pedagogy (Teaching)		0
Clarity		0

Figure 6-9 Language theme expanded (NVivo 12)

- Step 5: Define and name themes
 - The next step involved defining the themes. A mind map was created (using draw.io) to identify the relationship between the categories. Figure 6-10 displays the theme map. This was invaluable in defining the final themes of clarity, pedagogy and Content Topics/Subjects.

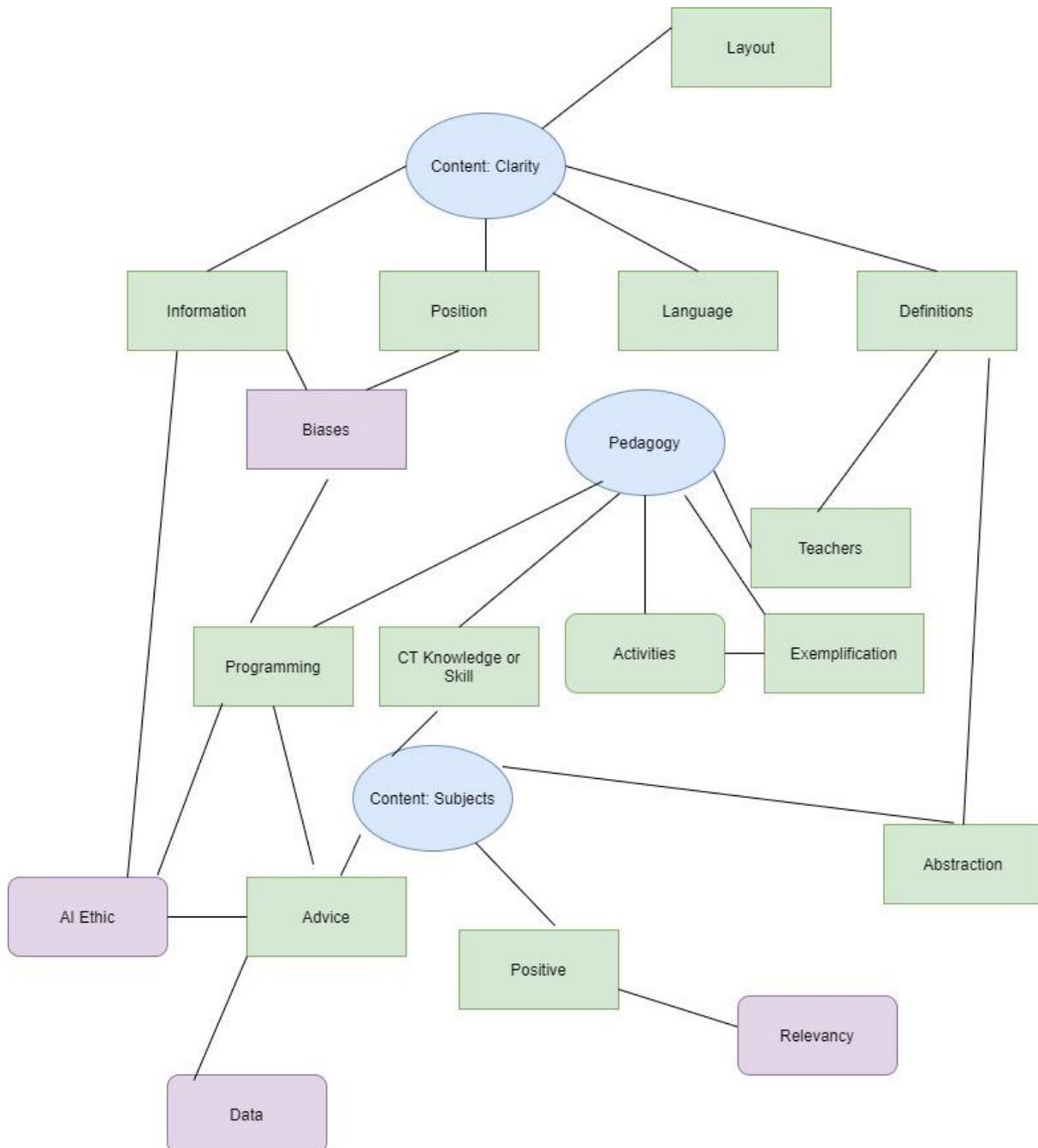


Figure 6-10 Mindmap of the Content Experts Themes

- Step Six: Write up analysis
 - The following sections concerns discussion on all three themes: ‘Clarity’, ‘Pedagogy’, and ‘Subjects’.

6.5.1.3.4 Clarity

‘we want to maximise interest, minimise misinformation and misconception’ (Expert 2).

With reference to the course content and its quality, ‘Clarity’ can be considered the most important theme. It overlapped with the other two themes: ‘Pedagogy’ and ‘Subjects’.

In their feedback, both experts referred to items that can be considered under the umbrella of clarity, for example, definitions, course layout, information, position on definitions, and language. Chapter 2 of this thesis outlines my position on how this study defines

Computation Thinking, specifically Selby and Woollard (2014) definition and the rationale for its use. I failed to do this in the course guide. I defined Computational Thinking in general terms as a problem-solving framework that uses concepts from Computer Science.

Whilst these concepts are explained, I do not set out my position ‘... super, super, ridiculously explicitly’ (Expert 2) and provide a rationale for same. Expert 2 put it as follows:

My concern would be the minefield of Computational Thinking. There’s a million definitions out there, and there are proponents, and there are opponents, and there are even leading people in computing education who have at different times completely dissed the idea of Computational Thinking almost going as far as Computational Thinking doesn’t even exist. All the way up to champions of Computational Thinking (Expert 2).

Thus, to answer a question about the relevance of topics and subjects in the course, teachers need to know my definition of Computational Thinking. This oversight was corrected for Version 2. On a similar note, it was recommended that I add clarity to the layout of guide by developing a curriculum map and placing it at the start of my document (Expert 1):

If you're asking me did you include relevant stuff; you did. But that sense of what was relevant and what was in and what was out wasn't entirely clear to me. Until I got to that point in looking through. (Expert 1)

Expert 1 highlighted the importance of clarity in language, particularly where the word had a specific meaning in Computer Science. This concern was in relation to two words: sorting and abstraction. During the piloting of Version 1 of the course, I introduced the pre-activity sorting of buttons. Expert 1 questioned the word 'sorting', as the activity resulted in students 'sorting' based on colour and the number of holes in the buttons. The Oxford University Dictionary (Lexico, 2021) define sort as: 'A category of things or people with a common feature; a type.' In Computer Science, the word sort means 'rearranging information into ascending or descending order by means of sortkeys' (Butterfield, Ngondi and Kerr, 2016). I changed the activity to be labelled as categorising buttons, and updated my notes to highlight the Computer Science meaning of sorting. There was also a discussion relating to my definition of abstraction. The course defines it as identifying and capturing essential details. I highlighted the reductionist and simplification aspects of abstraction as specified by Wing (2009). In Computer Science, there are two types of abstraction, modelling and encapsulation (The Open University, 2019). Wing emphasises the modelling approach, i.e. ignoring detail that is not of interest. The encapsulation approach is concerned with information hiding, which is very evident in programming. As I envisaged and was proved right, most students had not programmed, this type of abstraction is not referenced. I judged the cognitive load to be too high. My rationale for using this definition is now specified in the document.

Clarity of position and information are also discussed. Clarity of position is examined under the 'Pedagogy' theme, due to its considerable overlap with this theme. Clarity of information is the last sub-theme discussed. It concerns three areas of identified potential misinformation: 1) a Computational Thinking flow chart, 2) essential characteristics of algorithms, and 3) artificial intelligence. Expert 1 considered a flowchart I designed to

highlight Computational Thinking components could result in misinformation. He rightly implied that Computational Thinking components do not occur in sequence. This slide was redesigned to show Computational Thinking components as a circle.

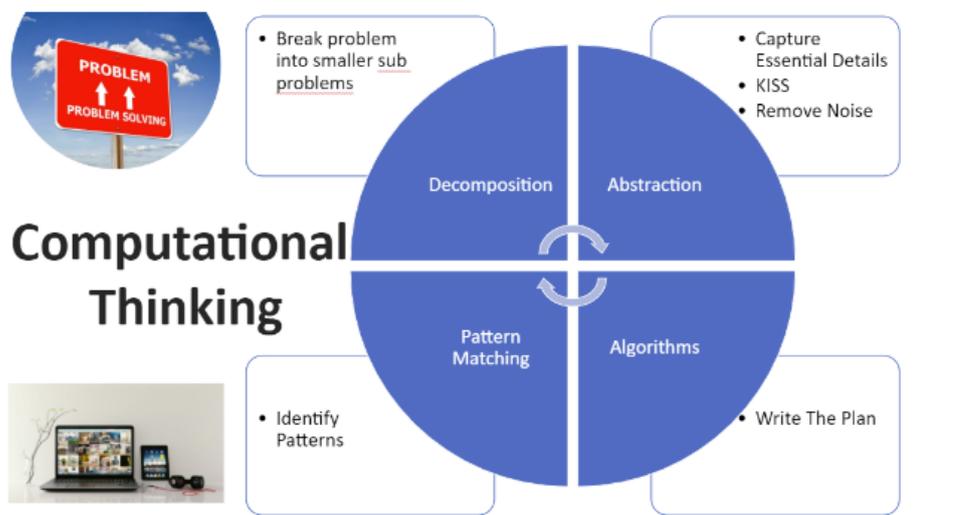
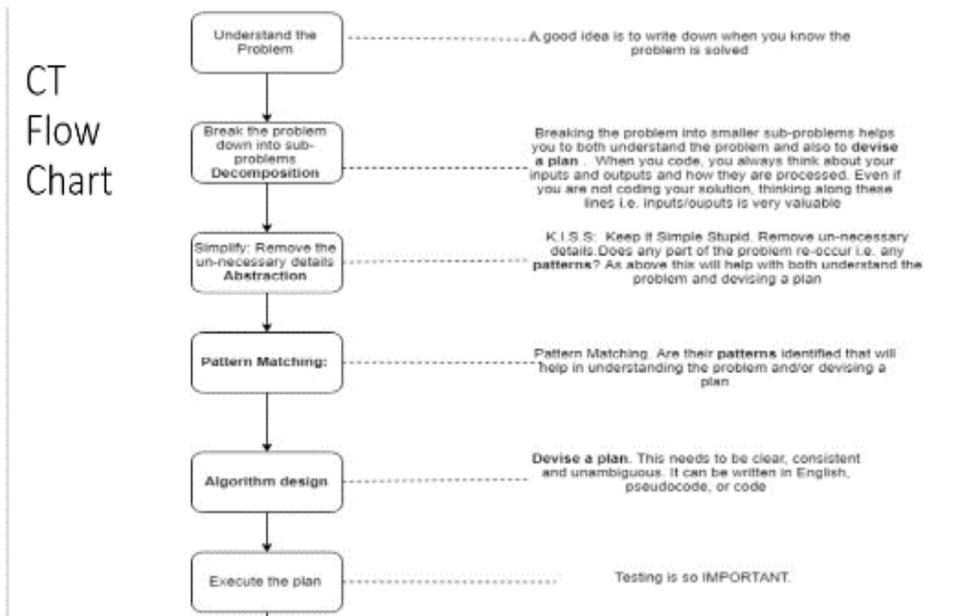


Figure 6-11 Computational Flow Chart replaced with CT components in a circle

Expert 1 also disagreed with my list of essential characteristics of algorithms. It was suggested that my programming bias was apparent in my view that algorithms can be considered as input, processing, output and that algorithms have a self-working aspect.



George Polya (1945) Book "How to Solve It" recommends four phases

- Understand the problem
- Devise a plan
- Carry out the plan
- Look back



Algorithms

Devise and Execute the Plan

- Step by step series of instructions
- Key words:
 - **Self-working**, i.e. The series of instructions always result in a specific result. Based on the input, we should know the output
 - **Unambiguous**
 - **Sequential**
 - **Finite**
 - Written in English, pseudocode or a programming language

Hamburger image (Gentgen) / GFDL (<http://www.gnu.org/copyleft/fdl.html>), CC BY 3.0 (<http://creativecommons.org/licenses/by-sa/3.0/>), CC BY-SA 3.0 (<https://creativecommons.org/licenses/by-sa/3.0/>) or GFDL (<http://www.gnu.org/copyleft/fdl.html>), via Wikimedia Commons



Input → Program → Output

EXPLAIN

Algorithms

Devise and Execute the Plan

- Step by step series of instructions
- Key words:
 - **Consistent**, i.e. The series of instructions always result in a specific result. Based on the input, we should know the output
 - **Unambiguous**
 - **Sequential**
 - **Finite**
 - Written in English, pseudocode or a programming language

George Polya (1945) Book "How to Solve It" recommends four phases

- Understand the problem
- Devise a plan
- Carry out the plan
- Look back

Input → Program → Output



Figure 6-12 Essential Characteristics of algorithms Slide, before and after change

Expert 1 posited that this is an old-fashioned view as Scratch allows parallel programming and its loop construct has a forever option. I did not take this advice on board. Reflection on the design of the course highlighted how the course builds cumulative knowledge. This is apparent with the introduction of ‘pre-activation’ activities. Thus I kept my original characteristics of the algorithm. I liked the simplicity of input, processing, output and I was not going to introduce concepts of parallel programming as the cognitive load would be too high. Preliminary results from the course pre-tests showed that most students did not know what the word algorithm meant (see 6.6.1.4.3.2). I did change the wording of an algorithm

having a characteristic of ‘self-working’ to ‘consistent’, as self-working may imply autonomy. Expert 2 highlighted how there might be a possible risk of misinformation regarding AI, which is dealt with in the Subjects theme.

6.5.1.3.5 Pedagogy

‘Pedagogy’ is defined as the method and practice of teaching (Merriam-Webster, no date). This theme had four main sub-themes of teacher/position, CT Knowledge or skill, activities and exemplification. The subtheme teacher/position was an important subtheme that illustrated the relationship between instructional design and teachers. This theme overlaps with the ‘Clarity’ theme:

I think that possibly the next biggest variable is the teacher themselves

the bottom line is you’re focused on the curriculum, and that’s legitimate and that’s what you need to make sure is the best. But that doesn’t mean that you can totally ignore the teacher part. So, I would look at that interface between you as a curriculum person and if this is successful, the many, many teachers who you’ll never know or never meet who will teach this. And you have the power and control to try to shape how they will deliver this. So, I think that it might be a good idea to take a little time and write ... This is a teacher’s guide to the curriculum, but you almost need like a preface, which is where this course is coming from and how this course should be delivered. (Expert 2)

Expert 2 point is valid when you consider the intervention is being piloted to develop certain characteristics. The instructional model of the course should be highlighted in conjunction with the content. Gooder *et al.* (2012, p. 48) emphasize the ‘inseparable link between curriculum and instruction’. They highlight how the American ‘Exploring Computer Science’ course's instructional paradigm was as crucial as the selected content. This point is relevant when I reflect on my first time teaching this course, I resorted back to lecturing (the inclusion of the pre-activation activities helped to involve the students more). Expert 1 concurred with this point:

The third question you’ve got is does the curriculum have the potential of developing a student’s understanding, yeah? And I just wanted to say briefly on that that I think

it does. And I think it does because you have so many activities for students in it. Yeah? And I urge you to, you know, do as many of those as you can to keep that at the forefront. You remember the teachers saying to you lecturing, you know. (Expert 1)

On the point of pedagogy, I was asked to consider my views on whether I saw

Computational Thinking as knowledge or skill:

So, being thoughtful here about your Transition Year students, do you want them to be able to solve problems by applying the skill of computational thinking or do you want them to understand computational thinking in the sense of knowing how it works, and that, I think, is ... I think you would do both. I don't think there's one or the other but if you are doing both, you'll have to get them onto doing some solving of some problems through computational thinking. And that's when we come back to programming, because that is the par excellence place to do it. And there is that debate going on out there, I think, which you've probably come across between should we teach programming to develop computational thinking or do we teach computational thinking so that they can be better programmers and problem solvers, you know? (Expert 1)

This point links to the clarity of my design and position, where I see Computational Thinking independent of programming but recognise that in Computer Science, Computational Thinking is most commonly conveyed through a programming language. This course positions itself as teaching Computational Thinking so that students can be better problem-solvers and programmers (with the programming to be introduced after Computational Thinking).

The last two subthemes in this theme dealt with exemplification. Once again, there is a link with clarity. Expert 1 stressed that when introducing theoretical ideas, I have to ensure that I do more and not less exemplification. The reason for this is his concern that students repeat a word related to Computational Thinking but are not able to link it, (although he acknowledged that he could see I was aware of this issue).

6.5.1.3.6 Subjects

The last theme is ‘Subjects’. The theme of ‘Clarity’ and ‘Pedagogy’ both emerged from codes that were previously categorised as Corrective Feedback. The ‘Subjects’ theme captured data that had previously been concerned with the Validation category. Both experts confirmed that the course contained relevant subject and contents, and adhered to the principle of Computational Thinking:

so the first, the first comment I’d make about that relevance question is I think everything you’ve got in there is relevant. packed full of really quite powerful stuff, and Okay, so does the content reflect the key principles of computational thinking, I believe it does, yes. (Expert 1)

I do. I was happy and impressed with the diversity of examples and video clips, slides, links. But particularly examples in nice little digestible but hopefully high impact ways of engaging students and tying everything together. I think it’s really nicely coherent. I find it hard to believe that students would be disengaged or bored. (Expert 2)

[Me] Am I missing any subjects? Am I missing any topics?

No, I don’t think so. I mean, frankly, there’s so much there, and that’s not a criticism. I think it’s a strength. Nothing jumped out at me as being missing, no. (Expert 2)

[Me] Do you think it has the potential for developing students understanding of computational thinking?

Yeah, definitely. I think that probably the strength of it is that it’s fast – or it seems fast. There’s a lot there, so it probably won’t be boring. It should be really engaging. It covers a lot of ground. But it seems like all of the few minutes that you spend on each topic really are well curated and designed and developed (Expert 2)

There was one topic that both experts had concerns with, and that was artificial intelligence. Expert 1 felt that as it is such an important topic, it deserves its own course. He was concerned that I might not do the topic justice in such a short time frame. He felt my input might tokenise this topic. I took on board his comments. Whilst I don’t have the bandwidth in my course to deal with ethics in other fields, I did re-write Unit 4 for Version 2 where I linked the ethics topic to the German Ethics code for Automated and Connected

Driving. I also added a TED Video that highlighted how code written for automation results in pre-programmed actions rather than reactive human ones. This change tied in with Expert 2 concerns. He felt there was a chance I might misinform students. This is in relation to my standard algorithms being defined as deterministic, but the AI components being discussed in a different way. I stimulated interest using memorable videos, e.g. the Google Duplex computer booking a hair appointment as a human.

you don't want them to think that computational thinking is algorithms and deterministic and logic and you can always figure out why something happened if you go back where it's... And then artificial intelligence is the total opposite, because who knows what's happening there? And it can be biased and oh gosh, I don't even know what's happening there. It's just a really... They're almost on two extreme ends of a giant spectrum. And I wouldn't want students to form any other opinions or misconceptions or whatever (Expert 2)

I also changed my course content to stress that humans still write AI code and programs. AI programs may vary and make new rules based on data, but these new rules are still within the parameters of over-arching rules written by humans.

6.5.1.3.7 Summary

The course content of Version 1 was validated by content experts in the field of Computer Science and Computational Thinking. Revisions were made to the course based on their recommendations (see Appendix I). The retrospective analysis of the content experts' data highlighted the importance of clarity in both the design and intervention. It covered multiple areas from definitions, layout, language, topics and pedagogy. Clarity has been validated as an essential instructional element of this course, as it has been shown to be linked to the quality of the course. During the piloting of the course, clarity was linked to student engagement.

6.5.1.4 *Content Revisions: Student Recommendations*

Revisions were also made to Version 1 of the course based on student recommendations.

This data was gathered from the second open-ended question on the 'End of Course'

questionnaire: ‘Do you have any specific recommendations for improving the course?’

Student answers were analysed using content analysis see 4.7.4.2.1, (for the steps in this process). The findings are presented per school but discussed collectively.

Table 6-9, 6-10, 6-11, and 6-12 display the results from this analysis. Examples from each captured category, and the frequency of mentions are displayed on the same row.

SCHOOL 1

Table 6-9 Results from the analysis of the open-ended question: Do you have any specific recommendations for improving the course? (N=14, School 1)

Category	Description	Examples	Frequency Column	
			Mentions	By Person
Class Timing	Answers related to the timing of the lessons	‘Make it about a week or two longer as I feel even that short amount of time should be beneficial to students’ ‘Try to shorten the lessons as a lot of info to take in at one time’	3	3
Real-World	Answers related to more activities in real-life situations	‘More activities that I can use in real-life situations’ ‘Linking what we learned to what we can do with it ourselves’	2	2
Computer Science	Answers relate mainly to the use of computers and the topic of programming. One answer related to the topic of A.I.	‘more time on pseudocode or even a simple language like python, as this may be many students’ only opportunity to learn this skill’ ‘Using the computer for some aspects’ ‘programming in a programming language’	12 5(programming) 6 (Use computers 1 (More advanced AI)	8
Positive No Change	Answers specified that students happy with course the way it is	‘No’ ‘Not Really I loved the course so much. It really sparked an interest in computers and logical thinking. Thank you so much’	2	2

Table 6-10 Results from the analysis of the open-ended question: Do you have any specific recommendations for improving the course? (N=22, School 2)

Category	Description	Examples	Frequency Column	
			Mentions	By Person
Programming	Answers relate to the topic of programming and computers	<ul style="list-style-type: none"> <input type="radio"/> 'I enjoyed the coding aspect, More coding would be appreciated' <input type="radio"/> longer on the coding, as it a lot to give in one session <input type="radio"/> Write code and test it on a computer <input type="radio"/> more coding <input type="radio"/> computer room practice programming 	5	5
Positive: No change	Answers specified that students happy with course the way it is	<ul style="list-style-type: none"> <input type="radio"/> Positive <input type="radio"/> No <input type="radio"/> None <input type="radio"/> NO, all round was a really good experience <input type="radio"/> 'not really' <input type="radio"/> Nope 	7	7
Other	Answers not matching the other categories and contained diverse answers from timing to activities and teaching materials	<ul style="list-style-type: none"> <input type="radio"/> Longer 'concepts easier to understand' <input type="radio"/> More Videos <input type="radio"/> 'this type of learning may not work if not everybody is as involved an <input type="radio"/> 'shorter periods may make it harder to get fully engaged in the subj <input type="radio"/> less powerpoint and more examples on activities <input type="radio"/> more teambuilding activities <input type="radio"/> too many activities, better balance of lectures, videos and activities <input type="radio"/> More real-world examples in more areas. <input type="radio"/> Feedback 	9	8

Table 6-11 Results from the analysis of the open-ended question: Do you have any specific recommendations for improving the course? (N=26, School 3)

Category	Description	Examples	Frequency Column	
			Mentions	By Person
Timing	Answers relate to the timing of the lessons and/or course	<ul style="list-style-type: none"> ● 'I wish course was longer' ● 'I would love the course to be longer' ● more time on coding difficult ● leave them with the problem and see how they do ● more than 5 weeks I love to continue it ● 'I think it could have went n longer as it sparked curiosity in me a 	6	6
Positive: No change	Answers specified that students happy with course the way it is	<ul style="list-style-type: none"> ● 'I really enjoyed the course' ● 'I think that the activities should stay the same, it brok ● 'To be honest I loved every moment of the course as I' ● 'No' 'fantastic, interesting and time went really quick' ● 'I personally don't think it needs to be improved' ● 'All in all it was really good and wouldn't change that ● 'Not really. I think the course was well timed and had j 	8	8
Content	Answers relate to the course content	<p>More activities 4, More videos 4, More concepts linked back to real-life 3, Simplify more 2, More collaboration 2, More individual work 1, More theory 1, Wanted learning off by heart 1</p>	17	14
Programming	Answers relate to the topic of programming a	<ul style="list-style-type: none"> ● 'go slower when we were going through how to write t ● 'more coding ' (on physical computer) ● I wish more programming ● more time on coding difficult ● 'more activities for pseudocode' ● more computer science, more coding, courses in colle ● 'didn't like final week' 'heavy with information' go slo 	7	6

Table 6-12 Results from the analysis of the open-ended question: Do you have any specific recommendations for improving the course? (N=8, School 4)

Category	Description	Example	Frequency Column	
			Mentions	By Person
Activities	Answers relate to activities	‘More activities’	2	3
No change	Answer specified that students indicated no change necessary	‘N/A’	1	1
Other	Answers did not fit into other categories	‘More smarties’ Two were empty	4	4
Content	Answer stated more examples	‘More work and examples’	1	1

The takeaways from the analysis of Q2 are that eighteen students (23%) wanted no change to the course, with the same amount wanting a change to week 5, the programming week. Six students (7.6%) wanted the course to be longer than five weeks, with one student wanting shorter lesson times. Five students (6.4%) recommended more real-world activities and examples, and four students recommended more videos as they felt they aided understanding. Based on numbers, changes to unit 5 can be considered one of the most important recommendations. Students’ reasons were manifold: they found it difficult, wanted more coding, and wanted to code on a computer. Teacher 1a in her interview, also alluded to students wanting to use a computer. In response to a question on the practicality of the course, she answered the following

Me: .. from a practicality point of view; did you feel it lended itself to the classroom setting, so it was in a class and not a computer room?

Teacher 1a: Oh, absolutely. Maybe that is something that was said to me by the students as we went along, 'when are we going to be using the computers?' So, I don't know if you could lead on from what you were doing into actually using the computers because that was a notion that the students had, when were they be going

to use the computers? I couldn't tell them whether you were going to or not. Do you know what I am talking about?

Student answers suggest that some students, specifically in School 1, believed that the course would involve the use of computers. In summary, students answers strongly indicate that the programming lesson needs more work. It needs to be simplified, as many students found it challenging. The course as it stands does not lend itself to more time being spent on programming, although introducing computers is an option to consider for week 5. The practicality of this action was discussed with teachers involved in the second prototype of the course. An advantage of unplugged activities, is that you can use the resources straight away (Morreale *et al.*, 2012).

Unit 5 was completely rewritten for Version 2. An overview of these changes can be seen in Appendix G. A summary is as follows: a warm-up puzzle was added, two videos were added to aid understanding, the min and max pseudocode activity was replaced with a pseudocode activity concerned with writing to ascertain if a student passed an exam. To show the relationship between this pseudocode and a 'real programming language', python code that solved the same problem is run on an online compiler.

6.6 Evaluations

Data collected during the piloting of Version 1 resulted in changes and revisions to the course and its subsequent instructional design. Data collected was also used for evaluations. The course was analysed to ascertain if Version 1 was effective, engaging, practical, and high quality. This data was also analysed to investigate what characteristics of the evolving instructional design aided same.

6.6.1 Effectiveness

Effectiveness is evaluated using five levels of data: four of Guskey's five levels of data for assessing professional development (2002, 2003, 2016) combined with students' reactions. Four levels of data were collected in this phase (see Table 6-13).

Table 6-13 Instruments used to gather data on effectiveness

Criteria Variable	Sub-Variable	Design Phase	Instrument
Effective Guskey Model	Student reactions	Version 1	Student Questionnaire after course
	Teacher reactions		Teacher interview Teacher journal
	Teacher Learning	Version 1	Interview
	Student Learning Outcomes	Version 1	Awareness of CT Learning goals:
Student Learning	Pre and post test Portfolio of students learning End of course questionnaire (students) Teacher interviews		

6.6.1.1 Effectiveness: Student Reactions

Students’ reactions to the course were collected using both qualitative and quantitative means. This was achieved with the ‘End of Course’ questionnaire, which consisted of both Likert statements and open-ended questions. This data was analysed in two stages, the Validation and Standard Analysis Stage. In the Validation Stage, the quantitative data was analysed using descriptive statistics. The open-ended questions were analysed using a deductive content analysis approach where lists were made based on students’ recommendations. In the ‘Standard Analysis’ stage, the open-ended questions were revisited and analysed using content analysis to gain greater insight into the students’ perspectives. To avoid repetition, only the results from the descriptive statistics and content analysis are presented. These findings are presented per school but discussed collectively.

6.6.1.1.1 Quantitative Data Student Reactions

The ‘End of Course’ questionnaire contained five items related to effectiveness. Using a five-point Likert scale, students were asked to rate their agreement with the following statements

- Overall, I would rate the of this course as outstanding. (best_quality1)

- Overall, I would recommend this course to other students. (best_quality2)
- This course was a waste of time. (guskey_satisfaction) (reversed)
- I am dissatisfied with the activities used in this course. (guskey_satisfaction_3) (reversed)
- I am very satisfied with the content of this course. (guskey_satisfaction_1)

Table 6-14 provides a summary of the number of students participating from each school with the number of students who were available to complete the ‘End of Course’ questionnaire. However, it should be noted that the results from School 4 represent only 36% of the class. Only twelve consent forms were returned, of which nine of these students filled out the ‘End of Course’ questionnaire.

Table 6-14 Summary Table of the number of students participating from each school

School	No. of Participants	Number of students with programming experience	No. filled out ‘End of Course’ questionnaire
1	16	6	14
2	28	17	22
3	30	10	26
4	25	1 (Note only demographic data on 6)	9

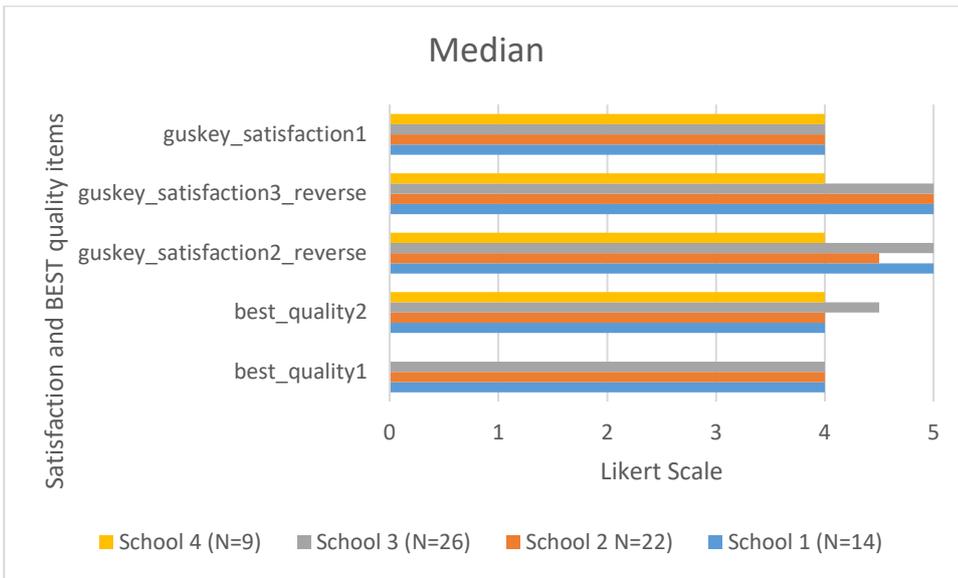


Figure 6-13 Median Values of the Satisfaction and BEST quality items

- *Note (BEST evaluation_1 item not on the questionnaire in school4)

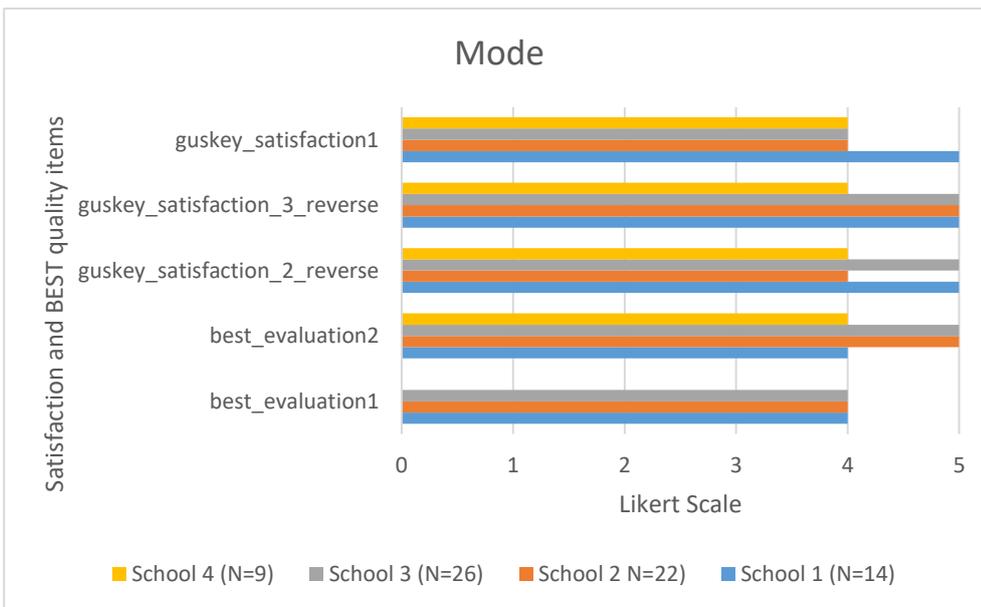


Figure 6-14 Mode Values of the Satisfaction and BEST quality items

Note School2 guskey_satisfaction_2_reverse is multi modal, 4 was smallest value

The mode and the median value for each item is either 4 or 5 (Figure 6-13, 6-14). This corresponds to Agree or Strongly Agree. It is apparent that the majority of students had a positive reaction to the course. In short, they found the quality of the course to be

outstanding, they would recommend it to others. They did not find it a waste of time and were satisfied with both the content and activities used in the course.

These findings are explored further with three open-ended qualitative questions. The answers to these questions were analysed using content analysis.

The questions were as follows: Q1 What are the strengths of this course? Q2 Do you have any specific recommendations for improving this course? Q3 What do you think you learned during the course?

The data generated from the first question, ‘What are the strengths of this course?’ are discussed in the next section, 6.6.1.1.2. This data was analysed using Content Analysis (see 4.7.4.2.1). The results provide an insight into students’ reactions to the course. Students’ recommendations have previously been discussed in Section 6.5.1.4. The findings for each school are displayed in tabular format.

6.6.1.1.2 Qualitative Data: Strength

The content categories, Activities, Set-Up and Learning, were identified as the course’s strengths by students in School 1.

Table 6-15 Results from the analysis of the open-ended question: What are the strengths of this course? (N=14, School 1)

Content	Description	Example	Frequency Column	
			Mentions	By Person
Activities	Answers mentioned the unplugged activities and students’ opinion of them	‘fun and interactive’ (Student A12) ‘I loved the engaging activities’ (students A1, A8) ‘peaked my interest’ (Student A1)	9	5
Set-Up	Answers related to the structure of the class and also how the course was different and unique	‘I liked how it was less like a class and more like a lecture, it was a bit more chilled out’(Student A4) ‘different’ ‘unique’ ‘teacher easy going’ ‘working with others was great’ (Student A4).	7	6

Learning	Answers related to learning a particular subject and its contents	'Helps you think outside the box more and think about it more logically' (Student A10) 'It is something involved in everybody's life although not many people knew much about it' (Student A3)	6	6
----------	---	---	---	---

Four content categories emerged from the data analysis from School 2: Pedagogy, Materials, Engagement and Learning. In this analysis, activities were classified as a sub-category of Pedagogy.

Table 6-16 Results from the analysis of the open-ended question: What are the strengths of this course? (N=22, School 2)

Content	Description	Example	Frequency Column	
			Mentions	By Person
Pedagogy	Answers related to the way the course was taught using activities, interactive, collaborative and the demonstrations	'taught well using media and with us doing activities such as the magic tricks' (Student N17) 'way topics are explained -- makes it easy to grasp' (Student N8)	7 (Activities) 22 (total)	6 13 (total)
Materials	Answers related to the course materials.	'good powerpoints on information' (Student N14) 'Images' (Student N11)	5	5
Engagement	Answers related to the course being engaging and interesting	'kept my attention for the entirety of the course' (Student N5) 'interesting and fun' (Student N10)	7	6
Learning	Answers related to learning a particular subject and its contents	'I found the course aided my logical thinking and helped me to explain my thought process' (Student N6) 'It provides an insight into things not talked about that much in the modern world' (Student N15)	9	7

Four content categories emerged from the data analysis from School 3: Pedagogy, Teacher, Content and Other. In this analysis, activities were also classified as a sub-category of Pedagogy.

Table 6-17 Results from the analysis of the open-ended question: What are the strengths of this course? (N=26, School 3)

Content	Description	Example	Frequency Column	
			Mentions	By Person
Pedagogy	Answers related to activities and the way the course was taught, i.e. interactive, collaborative and using demonstrations	'easy to stay focused as there was a lot of activities' (Student S12) 'the activities were the strength, they got everyone involved'(Student S13) 'activities... really helped me to understand and learn more'(Student S21) 'very interactive.. it put the learning into real-life situation''(Student S17) 'I love the group work'(Student S20) 'We were shown examples of what we were being told about' (Student S10)	23 (activities) 54	21 26
Teacher	Answers identified the teacher as a strength	'teacher is really nice, friendly and helpful' (Student S2) 'Colette was very engaging and fun to learn from' (Student S17)	7	7
Content	Answers related to the Course Content	'Personally I really enjoyed the algorithms and ethics section of the course' (Student S4) 'I also think the content of the classes was another strength as we learnt so many new things every week without all the information being too much to take in' (Student S6)	9	7
Other	Answers were mixed. They ranged from the course cheered them up, to the fact they enjoyed it	'Sometimes when I was fed up or in a sad mood, this course always cheered me up & I looked forward to it each week' (Student S26) 'Different use to teacher and PowerPoint ... this course ...involved ... use our brains' (Student S4)	4	4

Four content categories emerged from the data analysis from School 4: Pedagogy, Change in behaviour, Content and Other.

Table 6-18 Results from the analysis of the open-ended question: What are the strengths of this course? (N=9, School 4)

Content	Description	Example	Frequency Column	
			Mentions	By Person
Pedagogy	Answers related to activities and the way the course was	'the conversations between teacher and class and ideas' (Student C14) 'you learn a lot of activities' (Student C5)	5 (3 activities)	5

	taught, i.e. interactive, collaborative and using demonstrations	'It was very interactive which sets it apart from others' (Student C23)		
Change in Behaviour	Answers identified a change in their behaviour	'You start thinking more than you usually do' (Student C30)	1	1
Content	Answers related to the Course Content	'Learning about Computing' (Student C1)	1	1
Other	Question was unanswered		2	2

6.6.1.1.2.1 Discussion

The content categories of 'Activities', 'Pedagogy', 'Learning' and 'Course Content' were stated as a strength by most students. Whilst, there are overlapping elements and contexts between these categories, together they illustrate the course's strengths. Each of these content categories is discussed in detail in the following section.

6.6.1.1.2.2 Content Category: Activities

Activities were a category or subcategory that stood out during the analysis. 33% of the students in School 1, School 2, School 4 and 80% in School 3 mentioned them in their answers. Students considered them as a strength for a variety of reasons. These reasons included answers that correspond to the three facets of engagement: emotional, cognitive and behavioural (Fredricks *et al.*, 2011). Activities were described as: 'fun and interactive' (student A12), 'engaging' (students A1, A8), 'peaked my interest' (student A1). 'made the course more enjoyable' (Student S3). Hidi and Renniger (2006) highlight how positive experiences are important for developing personal interest in a subject and growth in students' belief in their proficiency. Several students linked this fun element to learning.

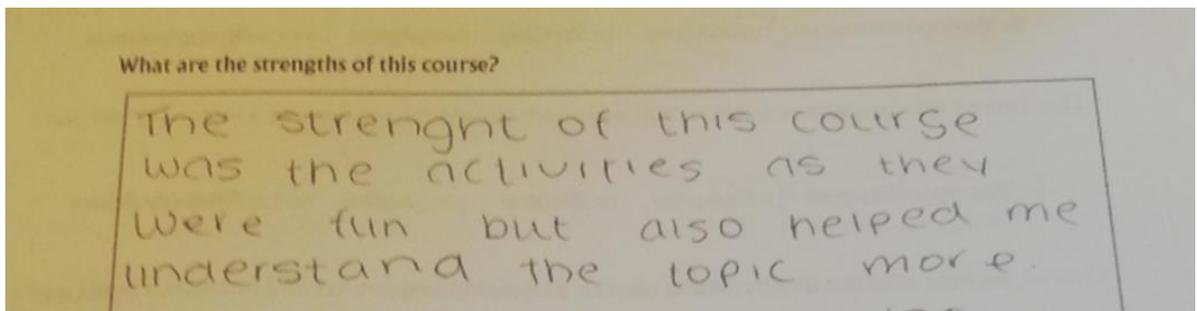


Figure 6-15 Student (S14) partial answer to: ‘What are the strengths of this course?’

A sample of students' comments includes: ‘I found them fun and when something is fun I usually learn more’ (Student A6). ‘I really enjoyed the practical games and explanations that we got to do before the theory as I felt it helped to explain the theory and information we were given. I found it made the course more enjoyable’ (Student A3). ‘The use of the activities to demonstrate all the concepts was the best possible way to conceptualise those concepts’ (Student A5). ‘Using activities to help us learn’ (Student A7).

Students also enjoyed the collaborative element of the activities, with Student A13 and Student N20 highlighting the inclusive element of the activities.

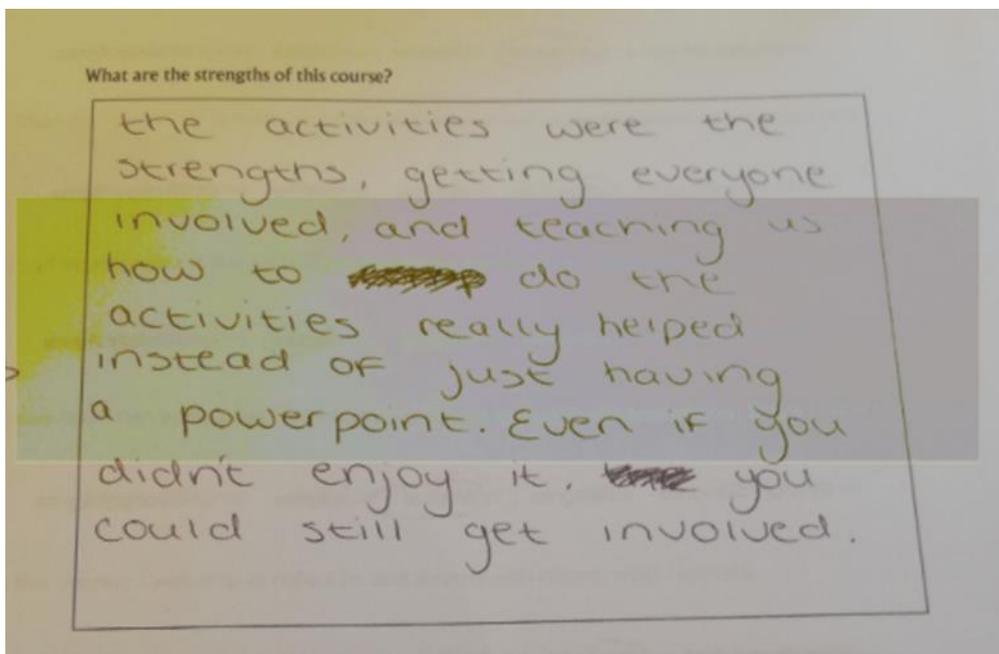


Figure 6-16 Student (A13): ‘What are the strengths of this course?’

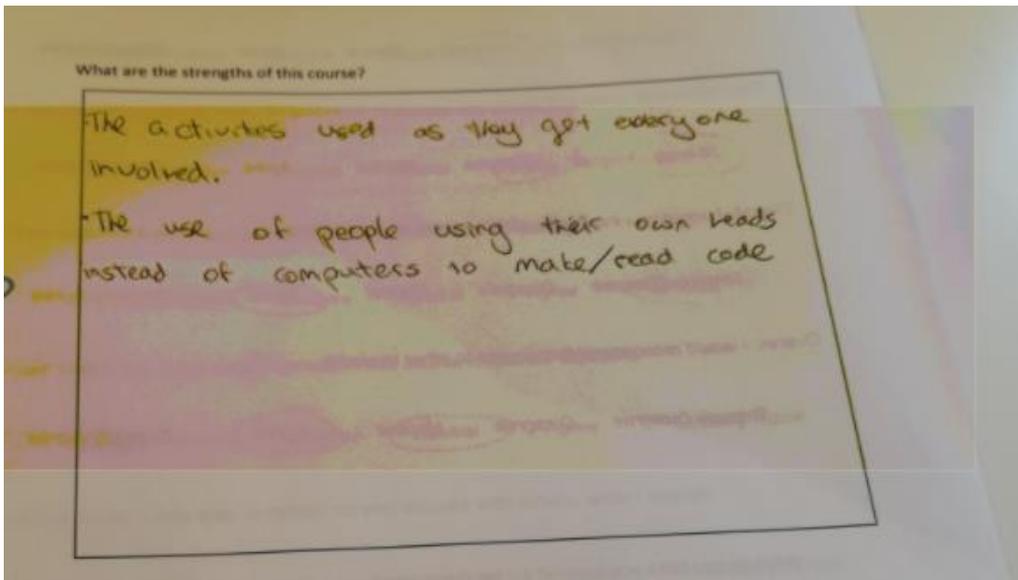


Figure 6-17 Student (N20) answer to: ‘What are the strengths of this course?’

Unplugged activities being reported as a strength of the course is very encouraging. The reason for this is manifold. Unplugged activities have their origins as part of outreach programmes and are geared at primary school students (Bell, Witten and Fellows, 1998; Rodriguez *et al.*, 2017). Bell and Vahrenhold (2018) highlight how there have been surprisingly few empirical studies about how Computer Science unplugged activities have been used in a regular classroom. There are also mixed reviews in the literature regarding its effectiveness. Feaster *et al.* (2011) study using Computer Science unplugged programs on high school students showed that their outreach programme had no impact on student attitudes towards Computer Science and their perceived content understanding. They conjectured the following reasons 1) that high school students do not find kinesthetic activities exciting, 2) the students were already involved in Computer Science programming classes, and thus believed they were already experienced programmers. The students in my PhD study saw the activities as a strength of the course. Their answers show that they had a positive experience of unplugged activities.

The students’ answers were also cross-validated with quantitative items from the ‘End of Course questionnaire’ related to activities.

6.6.1.1.2.3 Quantitative Data: Unplugged Activities

The ‘End of Course’ questionnaire contained five items related to unplugged activities. Using a five-point Likert scale, students were asked to rate their agreement with the following statements

- I am dissatisfied with the activities used in this course (g_learning2_reverse)
- I engaged in activities that helped me learn ideas or skills that were new and unfamiliar to me. (m_activation1)
- In this course, I was able to connect the activities to new ideas and skills I was learning. (m_activation3)
- The activities used in this course were helpful in learning. (unplugged_1)
- The activities helped increased my knowledge and skills in Computational Thinking. (unplugged_2)

The mode and median values for each of the five items per school are illustrated in Figure 6-18 and Figure 6-19.

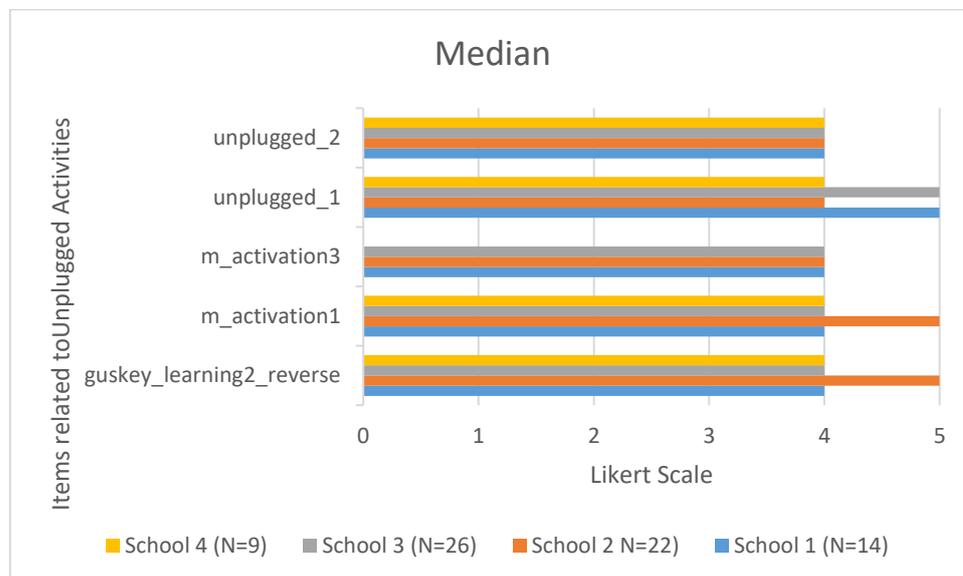


Figure 6-18 Median Values of items related to Unplugged Activities

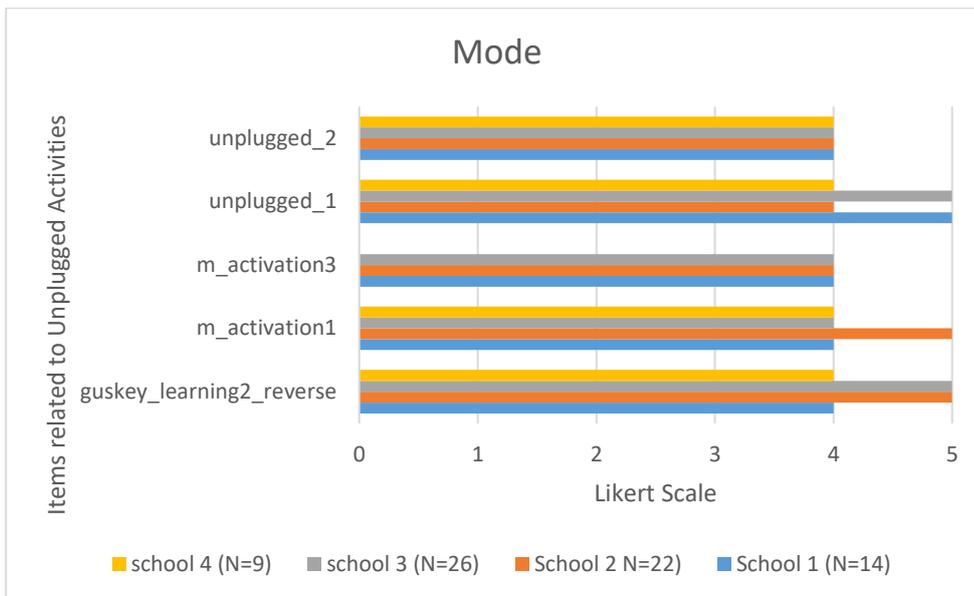


Figure 6-19 Mode Values of items related to Unplugged Activities

Note School1 guskey_learning_reverse is multi-modal, 4 was the smallest value

From the above bar charts (Figure 6-18, Figure 6-19), the data shows that the mode and the median value for each item is either 4 or 5. This corresponds to Agree or Strongly Agree. It is apparent that the majority of students had a positive reaction to the unplugged activities used in the course. They linked these activities to learning new concepts and skills.

Students also linked activities with pedagogy. Student N7 summation of the course strengths highlights these links. He states that ‘activities secure the learning’, ‘practical learning style’ and ‘problems that felt rewarding to overcome but not demotivating to fail’.

Pedagogy as a course strength is discussed in 6.6.1.1.2.4.

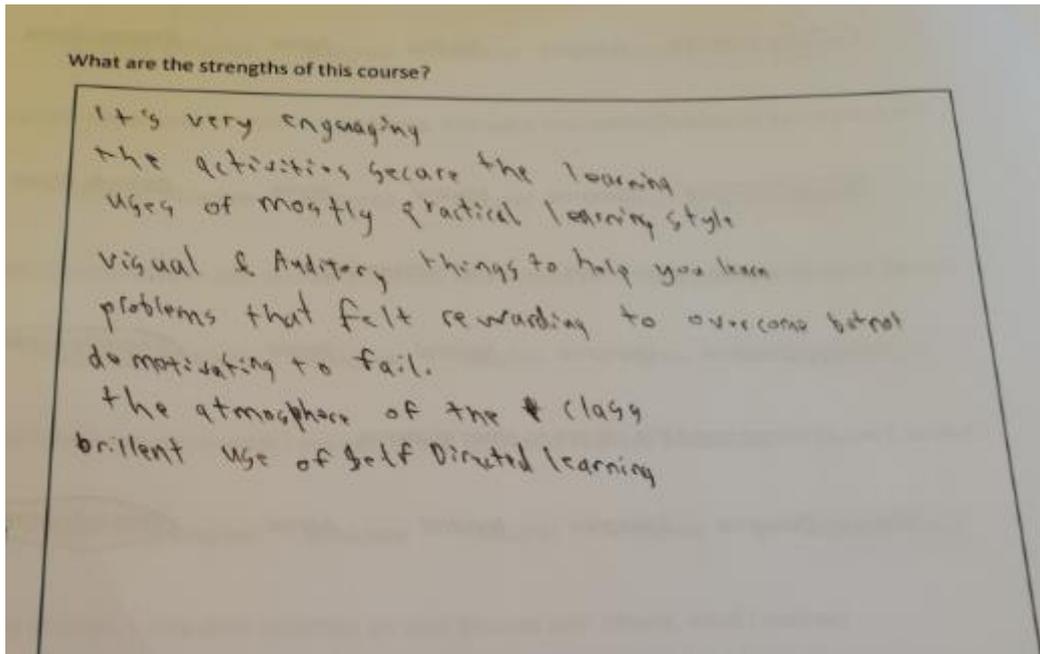


Figure 6-20 Student (N7) answer to: ‘What are the strengths of this course?’

6.6.1.1.2.4 Content Category: Pedagogy

Students also expressed the way the course was taught as a strength. Their answers categorised under this topic were diverse. Examples include the course's visual and auditory elements, class atmosphere, exemplification and collaboration. Figure 6-21 displays the in vivo answers coded from School 2 under this content category.

Strengths	0
Pedagogy	0
'way topics are explained -- makes it easy t	1
very interactive	1
'usage of practical learning style'	1
'slow and clear explanations'	1
reflection	1
'visual and auditory things to help you lear	1
brillant use of self-directed learning'	1
'breaks to evaluate'	1
'taught well using media and with us doin	1
'atmosphere of the class'	1
'showing examples of how coding was use	1
'use of people using their own heads inste	1
activities	2
Collaboration	3

Figure 6-21 In Vivo answers coded under the Pedagogy category for School 2

6.6.1.1.2.5 Quantitative Data: Demonstrations

Student answers related to pedagogy were cross-validated with two quantitative items on the ‘End of Course’questionnaire. The items were:

- Media used in this course (e.g. videos, websites, slides) were helpful in learning.
- My teacher gave examples of concepts that I was expected to learn.

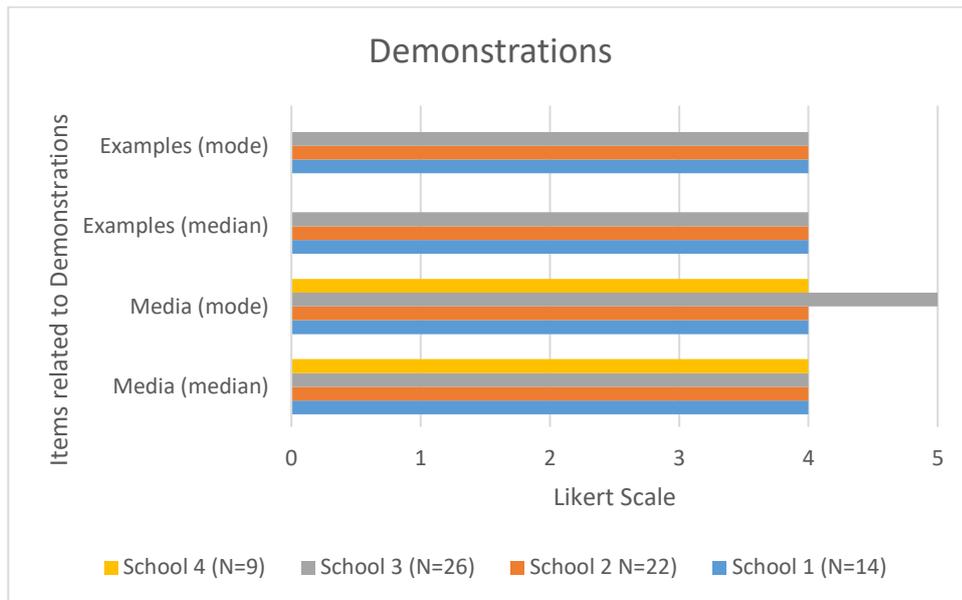


Figure 6-22 Mode and Median values for demonstration items per school.

The mode and median values for each of the two items per school were 4 or 5. These central tendency results indicate that most students were happy with both the materials used in the course and the examples used.

6.6.1.1.2.6 Content Category: Learning and Content

The content category ‘Learning’ was also stated as a strength of the course. Student Learning is discussed in depth in section 6.6.1.4. However, as it was also mentioned as a strength of the course by many students, it is briefly discussed here. Similar to the pedagogy category, student answers were diverse. In School 1, six out of the fourteen students mentioned learning as a course strength. Three of these answers referred to logic, and the other three to the application of the course to real-life situations. ‘Helps you think outside the box more and think about it more logically’ (Student A10). ‘Thinking logically was the biggest strength of the course because it shows how to make computers and AI work’ (Student A2). ‘It is something involved in everyday life, although not many people know about it’ (Student A3). ‘Algorithms is the strength of this course because they can help you with certain situations throughout your life’ (Student A9). In School 2, nine answers related to learning, with three specifically referring to coding and gaining an

insight into what it is about. In School 3, there was not a specific learning category but a content category where nine students highlighted the course content. Of these answers, the algorithms and ethics content was mentioned the most. 'I really enjoyed the algorithms and the Ethics section of the course. I'm usually very easily distracted and if I'm not interested in something, I don't participate well but I didn't find that happening, I really enjoyed this course' (Student S4).

6.6.1.1.2.7 Summary

Section 6.6.1.1.2 discussed findings related to student reactions to the course, specifically their opinion of its strength. The quantitative data generated from the 'End of Course' questionnaire highlighted students' positive reactions to the course whilst also validating the qualitative data. The analysis of the qualitative data explored the reasons why students were satisfied with the course. These answers serve to inform the instructional design as they validated both the use of activities and the course's current pedagogy, such as the use of videos, collaboration and reflection. Findings also indicated that students were engaged with the course. This has relevance regarding the content of the course. Axelson and Flick (2011) highlight that students become disengaged when tasks become either too complex or too easy. Students were engaged with both the content and activities. They also found the course learning to be a strength. These findings suggest that the course content is age-appropriate and relevant for students.

6.6.1.2 *Effectiveness Teachers Reactions and Learning*

Data concerned with teachers' learning and reactions to the course was collected by qualitative means: emails, journals and interviews. This data was previously analysed during the 'On the Fly' stage to initiate revisions to the course. These changes were validated during the 'Validation Stage' and are documented in section 6.5.1.1. The teachers' qualitative data were re-analysed in the Standard Stage using thematic analysis. Similar to before, the purpose here was to gain a deeper understanding of the data to

inform the intervention's instructional design. This analytic process was both inductive and deductive. I started the analysis in NVivo by creating five nodes: Validations, Corrective Feedback, Quality, Effectiveness and Engagement. Following Braun and Clark's six steps, four themes were generated from the data. They are 'Effectiveness', 'Engagement (how)', 'Content' and 'Pedagogy'. This section is concerned with the 'Effectiveness' theme.

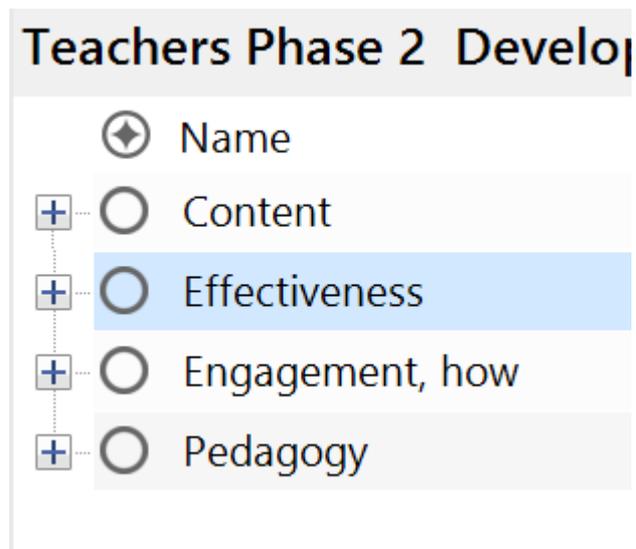


Figure 6-23 *The four themes generated from the analysis of the teacher data*

6.6.1.2.1 Teachers' Reactions

This section is concerned with teachers' reactions to the course. It also provides information regarding the practicality of the course for the classroom setting.

Teachers Phase 2 Developing Categories		
Name	Files	References
Content	1	1
Effectiveness	0	0
Set Up	0	0
Students	0	0
Teacher Learning Outcomes	3	4
Teacher Reactions	0	0
'enjoyed'	1	1
'intending to teach'	1	1
'Interesting' 'surprisingly i	3	11
Part of the class	1	1
'really good success'	1	1
recomendation	4	9
Stayed with you	2	2

Figure 6-24 Screenshot NVivo showing codes for Teacher Reactions

Teachers’ reactions to the course were captured with in vivo coding. The words used were enjoyed, engaged, relatable. thought provoking, interesting. A significant reaction, that was captured by three teachers was coded under the heading of ‘New and Different’. This reaction was notable as it highlighted a need the teachers saw for the course.

[Me] do you think it has a place in the curriculum or is it just for people with Computer Science or do you think there’s a...

[Teacher 1b] I think it has a place definitely. I think it's actually really important because it's not anywhere at the moment in terms of the computer studies that you do in school like they're learning Microsoft you know they're learning Word, they're learning just how to use a computer basically. So it doesn't exist at the moment and it's where the world is heading so I think it definitely needs to come in somewhere and give the boys an idea of kind of and girls an idea of what it is and what they can achieve from it or kind of and also just even giving them skills that they can use if they never even go near a computer again it gives them the skills that they can use in

different aspects of life, just life itself breaking stuff down into simpler things, you know that kind of way, yeah

The most common word used in connection with the course content is 'interesting'.

Teacher 2 expresses this best by stating that the course is 'surprisingly interesting ... so it was kind of surprising how I would have expected it to be a lot less interesting you know to explain the concepts of Computational Thinking' (Teacher 2). Teacher 1b echoes this statement by stating that with reference to the logic game: 'I found that so interesting and like I really wanted to get involved in that one'. With reference to the course in general, she states that:

'I found it really interesting all this stuff that you were teaching and I think the boys engaged on the same level, most of them, as me, so yeah I would think it would be like I can't think of a week that I didn't enjoy sitting there listening you know that kind of way, so I think it was really engaging, yeah' (Teacher 1b)

Teachers also gave their reactions to the practicality of the course. They referenced that they liked that the resources used were everyday items and agreed that the best length for a lesson should be eighty minutes. Teacher 3 explained that a forty-minute class goes by in a heartbeat, and it is only as you are getting into something that the bell goes. All teachers mentioned that with the new Junior Cycle, the demand for the computer room or Chromebooks was high due to the need for presentation software for Classroom-Based Assessments. Teachers also expressed the views that they felt students learned a lot. Teacher 2 observed how the learning and the processes could be applied to many different subjects. He also remarked on how on an Easter School trip, his students talked about the course and the ideas it related:

you know there's stuff with fantastic learning behind them but I know certainly the lads went away because I was away with them over the Easter there as well and they were mentioning you know there was very specific parts that they were thinking back to, like the card tricks and the cards were out, the lads were recreating it. (Teacher 2)

Teacher 4 remarked that the takeaway for her and her students is the fact that machines are programmed, they do not think for themselves. ‘Everyone thinks our machine are so smart and then you were explaining, no, they only know what you tell them to do and I think that is something that they definitely learned’ (Teacher 4).

[Me] Did you feel did they learnt anything through the activities they engaged in?

[Teacher 4] I think the way it was done, as you said, for example the algorithms thing and drawing the smiley face, like how does the computer know what to do. Nobody ever really thinks about that.

6.6.1.2.2 Summary

Teachers’ reactions to the course were positive. They found the content interesting, uniformly agreed that the instructional design and materials were suited for a classroom setting and that, time-wise, the eighty minutes lesson worked best.

6.6.1.3 Teachers’ Learning

This theme is concerned with one of Guskey’s four levels of data to ascertain the effectiveness of a course, participant learning. To help to ascertain this information Guskey (2002) poses the question: ‘Did participants acquire the required learning and skills?’ For the purpose of this course, the question is ‘Did teachers acquire the knowledge to teach this course?’

All teachers agreed that they felt confident teaching the course. Teacher 1b who had no previous experience, stated she would feel comfortable teaching but would need help with the programming lesson and with the ChatBot activity:

definitely would feel comfortable teaching it, I just would be really honest with them and say, ‘Look I don’t know everything, it’s something that we’re both – you know we can both look into together’, and all this kind of stuff you know but in terms of the actual course itself, yeah, as you said, there was only I think one lesson wasn’t it, that I said I’d want more on. (Teacher 1b)

The theme of ‘Teacher Learning’ is explored further in Chapter 7, where data concerned with teachers’ experience with the workshop and teaching this course is analysed.

Data captured by this theme ‘Teacher Learning’ served to validate a requirement from the ‘On the Fly’ stage to scaffold more the ChatBot activity and simplify the programming lesson.

6.6.1.4 Effectiveness: Student Learning Outcome

This section is concerned with one of Guskey’s four levels of data to ascertain the course’s effectiveness: student learning outcomes. To help determine this information, Guskey (2002) poses the question: what was the impact on students? Students’ reactions to the course have previously been discussed in section 6.6, this section is concerned with establishing their perspectives on what they learned during the course. This data was gathered from two Likert items and the third open-ended question on the ‘End of Course’ questionnaire. Student answers also served to inform the instructional design and the content of the course.

6.6.1.4.1 Quantitative Data

Using a five-point Likert scale, students were asked to rate their agreement with the following two statements

- I did not learn a lot in this course. (guskey_learning2_reverse)
- Compared to what I knew before I took this course, I learned a lot. (g_learning1)

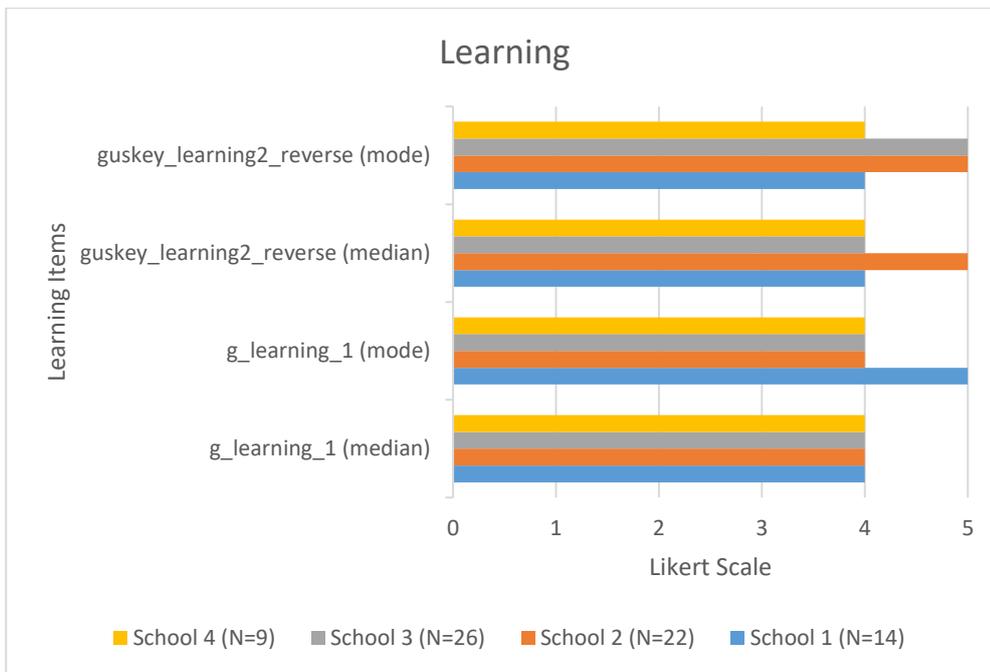


Figure 6-25 Mode and Median values for learning items per school.

- Note in School 4 students g_learning1 was ‘Compared to what I knew about Computational Thinking before I took the course, I learned a lot.’

As shown in the above barchart (Figure 6-25), the mode and the median results indicate that most students felt that their learning progressed during the course. These results validate the findings depicted in Figure 6-18 and Figure 6-19 which confirmed that students perceived the activities used in the course increased their knowledge and skills of Computational Thinking.

6.6.1.4.2 Qualitative Data

Student learning progress is explored further with the open-ended qualitative question: What do you think you learned during the course? Similar to before, each school's findings are tabulated before they are discussed as part of the analysis. Students gave a wide range of answers that were varied in detail, with some being quite informative.

Table 6-19 Results from the analysis of the open-ended question: ‘What do you think you learned during the course?’ (N=14, School 1)

Content	Description	Example	Frequency Column	
			Mentions	By Person
Logic	Answers related to the topic of Logic	‘I learned how to think more logically’(Student A2) ‘Overall what I found to be the more important and prevalent lesson in this course was the idea of breaking logic down into step by step sequence’(Student A5)	4	4
AI	Answers related to the topic of Artificial Intelligence or Ethics	‘ethics of programming self-driving cars’ (Student A3) ‘about how artificial intelligence and computers can be used in our everyday life’	5	5
Real-life	Answers mention relating skills to real-world	‘I have learned how I can use these skills in the real-world’ (Student A6) ‘I learned how to solve algorithms which are good for real-life situation such as maths exams’ (Student A9)	3	3
Computer Science and Computers	Answers relate to the topic of Computer Science and Computers	‘I learned a lot more about the theory of computer science which I am very happy about’ (Student A4) ‘learned about how computer work e.g. how they function and come out with solutions’ (Student A7)	2	2
CT and its components	Answers referred to CT and decomposition	‘better understanding of Computational Thinking’ (Student A3) ‘learned many definitions to do with Computational Thinking(Student A13) ‘decomposition’ (Student A11)	7	7
Algorithms	Answers relate to the topic of algorithms	‘create algorithms to solve problems’ (Student A2) ‘solve algorithms’ (Student A9) ‘being more specific in instruction given can be beneficial’ (Student A8)	7	6

Table 6-20 Results from the analysis of the open-ended question: What do you think you learned

during the course? (N=22, School 2)

Content	Description	Example	Frequency Column	
			Mentions	By Person
Algorithms	Answers mention Algorithms	‘what an algorithm is’ (Student N4) ‘How an algorithm works’ (Student N5)	5	5
Ethics	Answers mention Ethics	‘ethical side of AI and possible dangers’ (Student N11) ‘ethics of computer programming’ (Student N7) ‘morality, AI and automation’ (Student N18)	6	6
Problem Solving	Answers refers to problem solving	‘analyse problems’ (Student N1) ‘approach a problem’ (Student N4) ‘I learned different ways of looking and solving a problem’ (Student N10)	3	3
Logic	Answers refer to logical thinking	‘logical reasoning’ (Student N7) ‘logical understanding’ (Student N22)	5	4
Computer Science and Computers	Answers relate to the topic of Computer Science and Computers	‘basics of how a computer works’ (Student N16) ‘think like a computer and see how a computer works’ (Student N20) ‘how a computer thinks’ ‘how you would tell it what to do’ (Student N8) ‘how a computer processes things’ (Student N5)	4	4
CT and its components	Answers referred to CT and its component parts	‘I learned a lot about Computational Thinking and how to utilise this type of thinking in coding and programming code’ (Student N16) ‘How CT can be used and applied to everyday problems’ (Student N11) ‘how to break down complicated problems into smaller less complex problems’ (student N22)	9	7
Coding	Answers relate to programming	‘I learnt about coding and I now think about coding and how	16	15

		computers think differently’ (Student N15) ‘steps involved in coding (Student N1) ‘I was exposed to coding for the first time’ (Student N17) ‘caesar cypher’ (Studenty N12)		
--	--	---	--	--

Table 6-21 Results from the analysis of the open-ended question: What do you think you learned during the course? (N=26, School 3)

Content	Description	Example	Frequency Column	
			Mentions	By Person
AI and ethics	Answers mention AI and ethics	I learned about AI and its use in our world today e.g. google assistant (Student S6)	8	7
Algorithms	Answers mention Algorithms	‘I learned about algorithms and about how a computer will blindly follow a set of instructions, hence why you have to be very specific about what you want a program to do’ (Student S6) ‘I learned a lot more about how computers work and how they blindly follow instructions in coding and programming’ (Student S22)	12	11
Activities	Answers refer to the activities and the learning that they were devised to promote	‘binary Search’ (student S6,S5,S10) ‘I also learnt about when trying to guess something, e.g. a letter, the fastest way is to half it and then you find out if it is before or after. Half it again and so on’ (Student S6) Spell checker ‘the way people misspell word’ (Student S25) ‘I really enjoyed writing our own chatbot as it really helped me to understand how they worked’ (Student S8)	11	10
Problem Solving	Answers relate to the problem-solving aspect of Computer Science	‘I learned how I needed to think and plan out problems before just rushing into my first idea’ (Student S12)	11	10

		‘I learned how to think more logically and efficiently in order to solve puzzles and problems’ (Student S5) I learned how to break down a question the way a computer does and focus on the small things first (Student S7)		
Computers	Answers relate to the topic of Computer Science and Computers	‘importance of technology’ (Student S2) ‘how computers work and get their information’ (Student S4) ‘computers are stupid’ ‘get all of their knowledge from humans’ (Student S23) ‘how computers cannot think for themselves’ ‘you have to programme them’ (Student S23)	13	10
CT and its components	Answers referred to CT and its component parts	‘I also learned how to decompose, abstract and pattern match. Decomposition was really useful with pseudocode’ (Student S5) ‘Firstly one of the main things that I learnt was about problem solving and the steps that need to be taken including decomposition, abstraction and execution. I will be able to use this knowledge going forward’ (Student S6)	11	11
Coding	Answers relate to Coding	‘I learned a lot of new computer skills such as coding, which I have always wondered what that was’ (Student S13)	4	4

Table 6-22 Results from the analysis of the open-ended question: What do you think you learned during the course? (N=9, School 4)

Content	Description	Example	Frequency Column	
			Mentions	By Person
Problem Solving	Answers relate to the problem-solving aspect of Computer Science	‘solving problems’ ‘learn problem solving skills’ ‘working together and communication skills to help solve problems’	3	3

Logic and Algorithms	Answers related to the topic of Logic and algorithms	‘How to use logic how specific you need to be for a computer to carry out an activity’ ‘I learned how specific we must be when instructing computers, and how it follows a pattern of rules’	2	2
Content	Answers relate to course content	AI Tech + mind games	2	2
Empty	Question not answered		2	2

6.6.1.4.2.1 Discussion

An overall learning goal for this course was that students would gain an insight into the problem-solving framework of Computational Thinking. Under this umbrella, students were introduced to a range of diverse and memorable Computer Science topics, such as decomposition, pattern matching, abstraction, algorithms, ethics, artificial intelligence and the fundamentals of programming. In answering the question: ‘What do you think you learned during the course?’ The sixty-nine students who answered this question provided a range of answers. The most mentioned topics regarding their learning were Computational Thinking (26 students, 38%), with another sixteen mentioning problem-solving. Twenty-three students said algorithms, nineteen students stated coding, eighteen students referred to ethics and sixteen to Computer Science.

Over half of the answers that mentioned Computational Thinking were very detailed, illustrating that students understood what Computational Thinking is, what it is used for and its components. Their answers related Computational Thinking to both Computer Science but also generic problem solving and logical thinking. ‘I learned how I needed to think and plan out problems before just rushing into my first idea’ (Student S12). ‘I found

this course aided my logical thinking and helped me to explain my thought processes’ (Student N6). The course provided students with a language to articulate their thinking but also skills that they can use in different subjects: ‘I have learned how I can use these skills in the real-world’ (Student A6). ‘I learned how to solve algorithms which are good for real-life situation such as maths exams’ (Student A9)

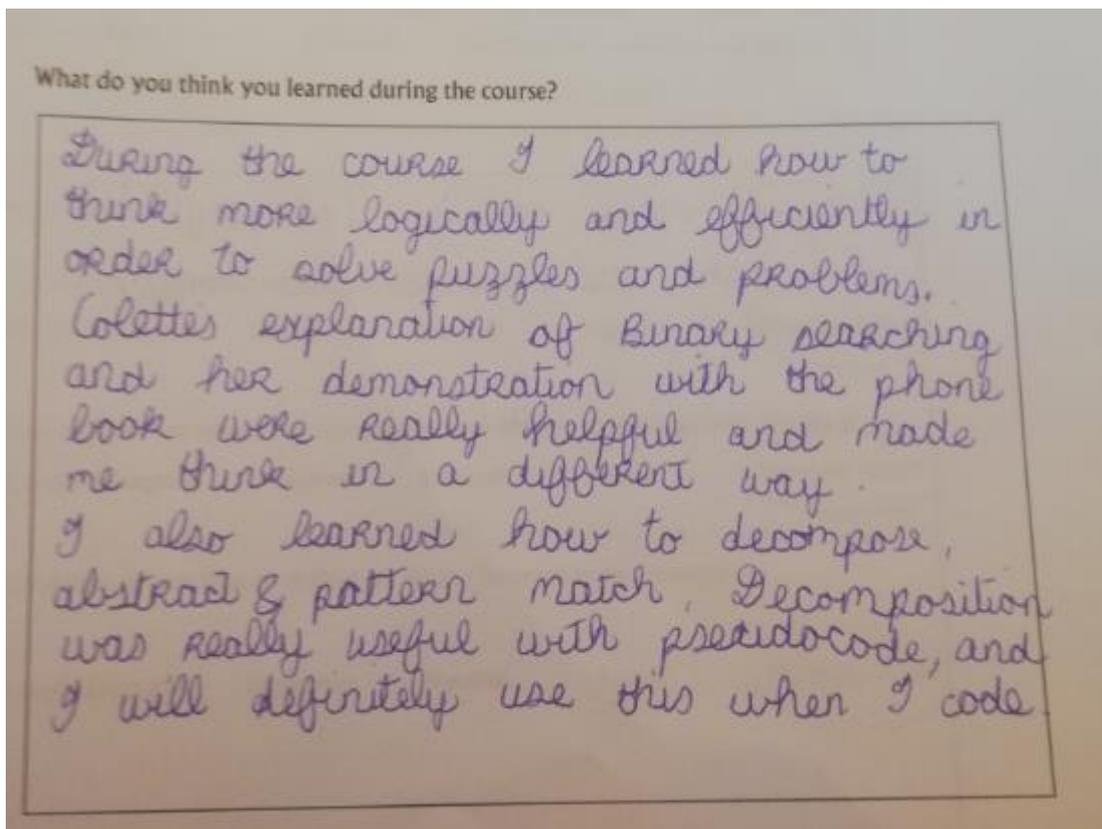


Figure 6-26 Student S5 answer to ‘What do you think you learned during the course?’

The surprising answers were those that highlighted how much the students learned even though they have no interest in Computational Thinking, or found some topics boring.

‘Even though I don’t have an interest in computers. I still learned and understood computer thinking’ (Student S13)

What do you think you learned during the course?

Before this course I did not know anything about computational thinking, so I learned a lot of new things. I learned how to figure stuff out using the six concepts such as abstraction, and also pattern and problem solving. I did find it harder though, as this is a topic I am not interested in.

Figure 6-27 Student S3 answer to 'What do you think you learned during the course?'

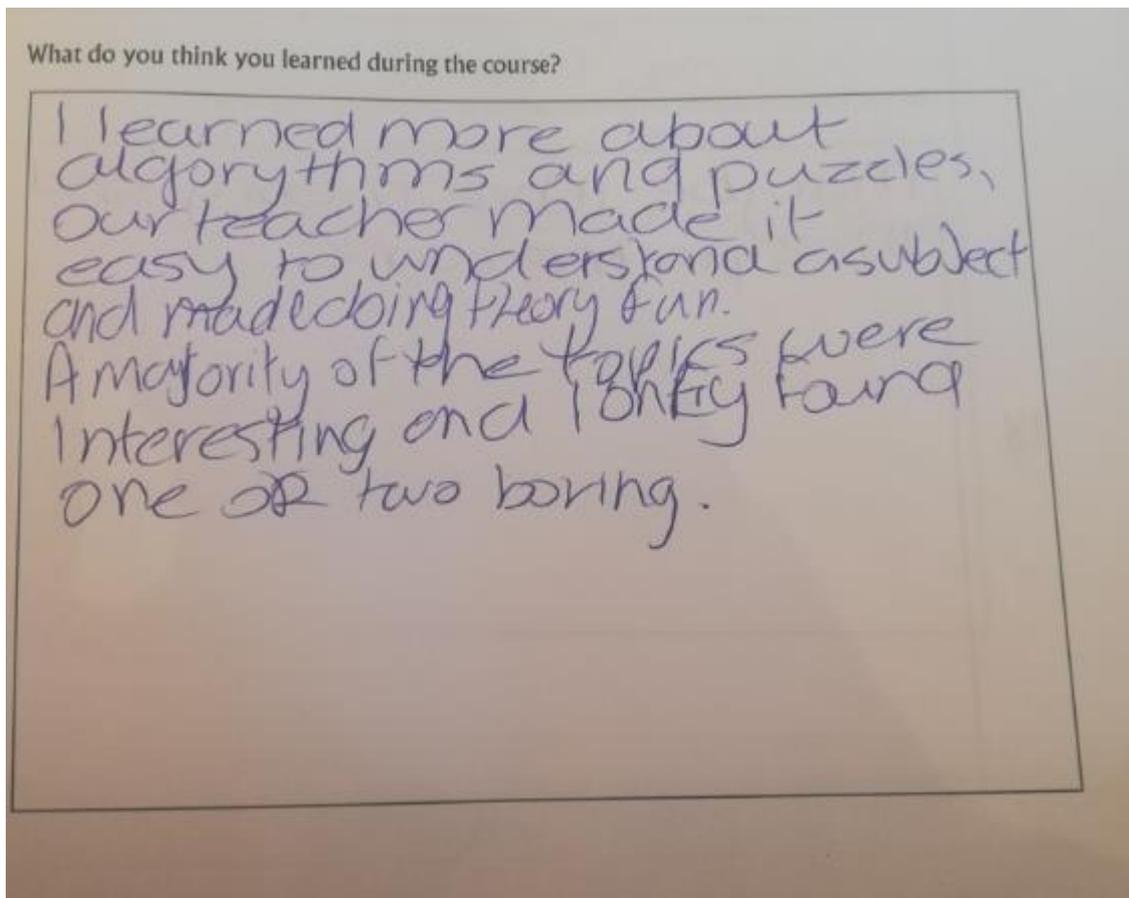


Figure 6-28 Student S18 answer to 'What do you think you learned during the course?'

Students also mentioned that they learned about algorithms. These results were encouraging as 70% of students scored Unsatisfactory in their pre-test on their understanding of algorithms (see section 6.6.1.4.3). The answers given were mixed in that half the answers were vague, only mentioning the word algorithm. However, the other half were detailed, highlighting how students biggest takeaway with reference to algorithms were the specificity of the instructions, and how a computer blindly follows them. 'I learned that computers are very stupid and that if you are trying to make it do something. You have to type everything exactly how it is that the program will run through it properly' (Student S5). 'When writing instructions to draw the emoji I learned how important it is to not miss any steps or it would mess everything up' (Student S21). Programming was the most common category in School 2, mentioned by fifteen of the twenty two students that filled out the form. The two answers that stood out from School 2 and School3, were in

relation to it being their first time exposed to programming. 'I was exposed to coding for the first time' (Student N17)

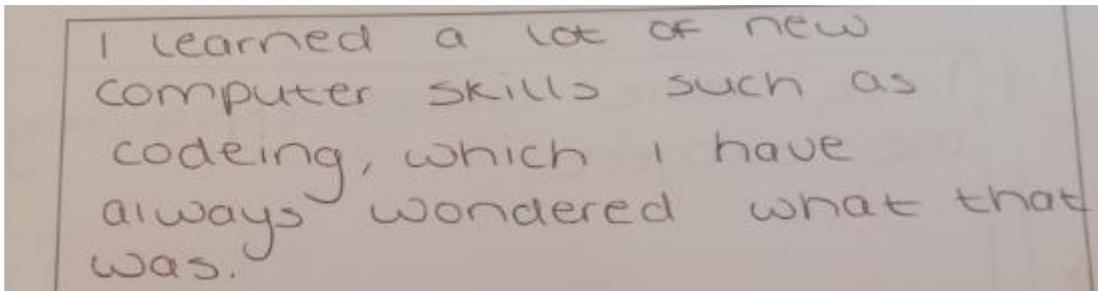


Figure 6-29 Student N13 answer to 'What do you think you learned during the course?'

Finally, one student who was already quite knowledgeable in this subject eluded to the fact that he learned Computer Science theory.

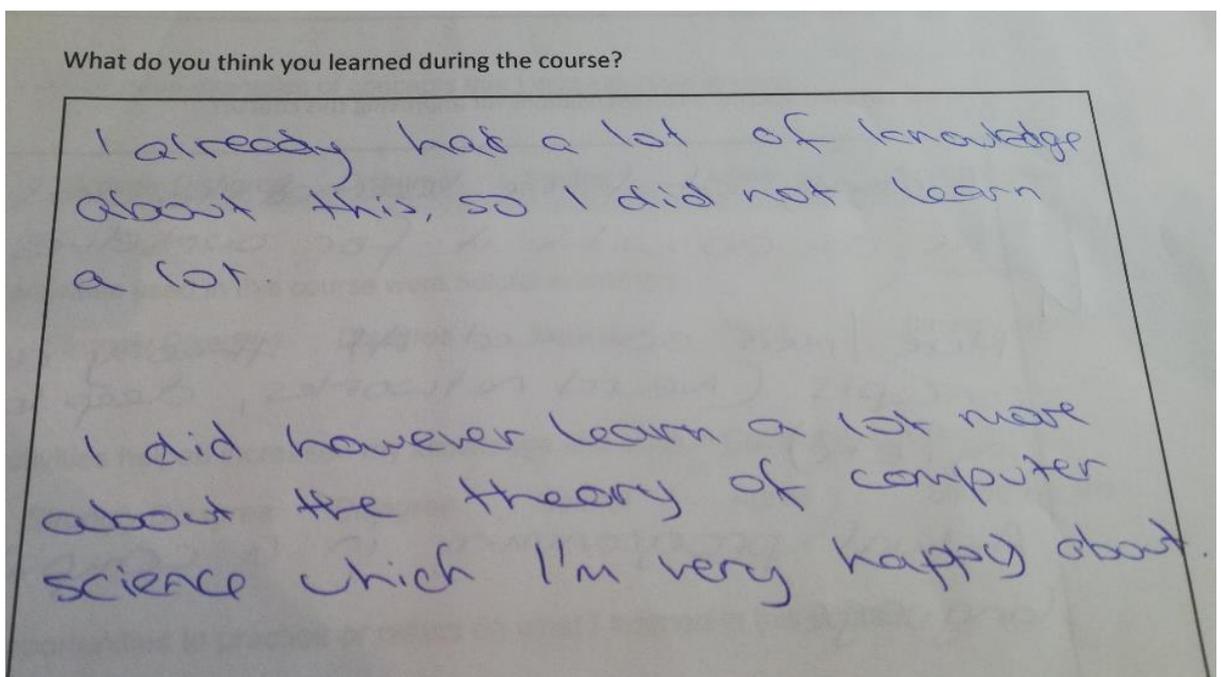


Figure 6-30 Student A4 answer to 'What do you think you learned during the course?'

6.6.1.4.3 Student assessment pre and post results

Pre and post-tests were issued at the start and end of the course. The pre-test consisted of six open-ended questions (and three demographic questions). Their purpose was manifold.

They ascertained students' awareness of the following topics, Computational Thinking, Artificial Intelligence and Algorithms. These topics were judged to be essential as 1) they play an important role in the course, and 2) there are challenges connected with their teaching, definitions and understanding (Sentance and Csizmadia, 2017; Holmes, Bialik and Fadel, 2019). The tests also served to inform the course's instructional design by highlighting misconceptions or instructional difficulties. They provided data to help answer Guskey's question on students' learning outcomes (Guskey, 2002). Finally, they verified the course content by confirming that students did not already know this information before the courses started. Two questions on the tests were as follows:

Pre-Test

Q1

- a) What do you think Computational Thinking might mean?
- b) Can you describe any situations/problems that might use Computational Thinking?

Q3

- a) Explain what you understand by algorithms?
- b) Give or guess some examples of algorithms.

Post-Test

Q1

- a) What is Computational Thinking?
- b) Describe in simple terms how would you use some (or all) the components of Computational Thinking to help solve the following problem, -- Creating a study timetable

Q3

- a) Explain what you understand by algorithms?
- b) Give or guess some examples of algorithms.

In the post test, students were also asked two questions about ethics and logical thinking.

Q1 and Q3 (pre and post test) were analysed and displayed using the steps described in Lamprou and Repenning (2018) study, which was based on content analysis. The responses for each of the above three questions were grouped according to several categories. The categories were created using the students' own answers. Student's answers were also graded using a rubric design influenced by Rodriguez *et al*'s (2017) assessment of unplugged activities. His rubric had three categories: unsatisfactory, partial

proficient and proficient. The students' responses were categorised as belonging to one of these categories based on the definition and content used in the course see (Appendix J).

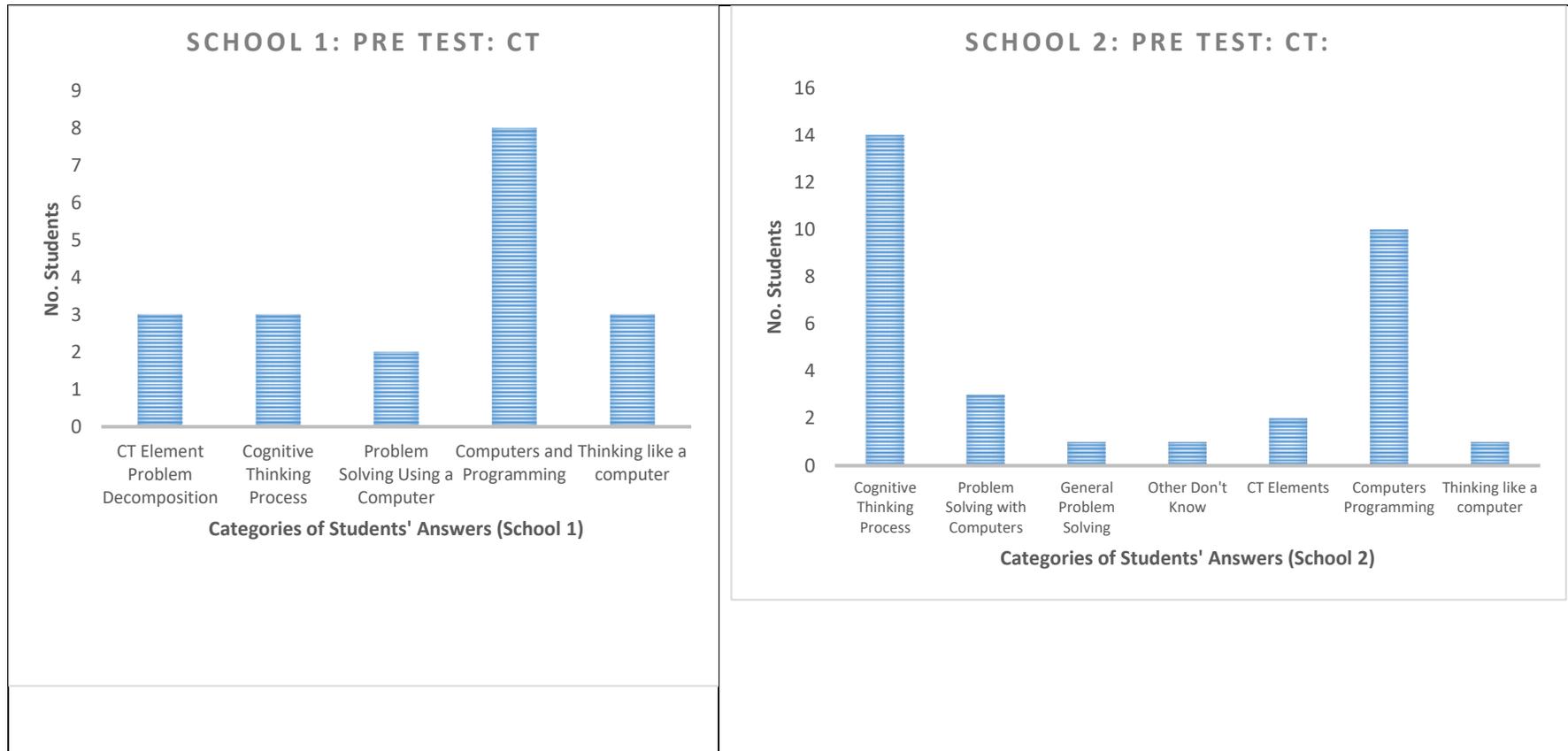
The following section describes the results from each school, on questions related to Computational Thinking and algorithms pre and post test. Each question and school is discussed individually.

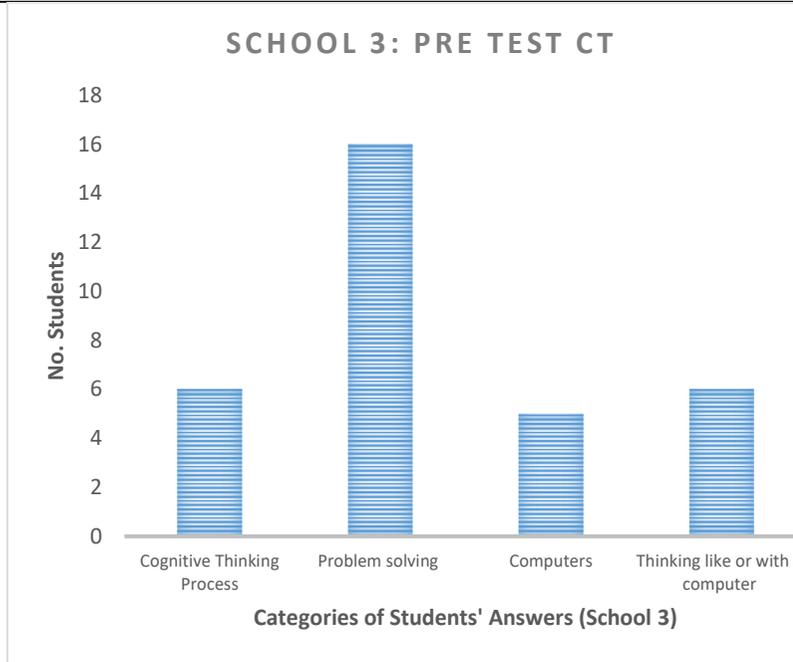
6.6.1.4.3.1 What do you think Computational Thinking might mean? Analysis

The following section displays six bar charts. These depict the categories and frequency per school of answers for Q1: What do you think Computational Thinking might mean?

The categories were developed per school from students' answers. The charts are displayed for each school, pre and post-test. This is followed by Table 6-25 and Table 6-26 which show the results from the content analysis of School 1 with students' examples and grades for the question.

Table 6-23 Bar charts depicting the category of answers and their frequency for CT Q pre test.





In School 1, fifteen students took the pre-test, resulting in eighteen unique answers, six students had previous programming experience. The majority of responses in School 1 associated Computational Thinking with Programming or Computer Science.

A similar number, ten, associated CT with programming/ computers in School 2. However the most common answer in School 2 was the association of CT with a cognitive thinking process such as logical thinking.

In School 3, the majority of answers associated Computational Thinking with problem-solving. This was initially surprising. However, the consent forms stated the following. ‘Computational Thinking is the term given to a set of skills that originate in Computer Science that can help you solve problems’. It can be surmised that this had an effect on students’ initial understanding of CT, especially as one of the answers was identical to my definition. Their answers to part b) Can you describe any situations/problems that might use Computational Thinking? highlighted that they did not understand Computational Thinking. Their answers concerned fixing broken computers: ‘the breaking down of a computer’ (Student 73)

Table 6-24 Bar charts depicting the category of answers and their frequency for CT Q post test.

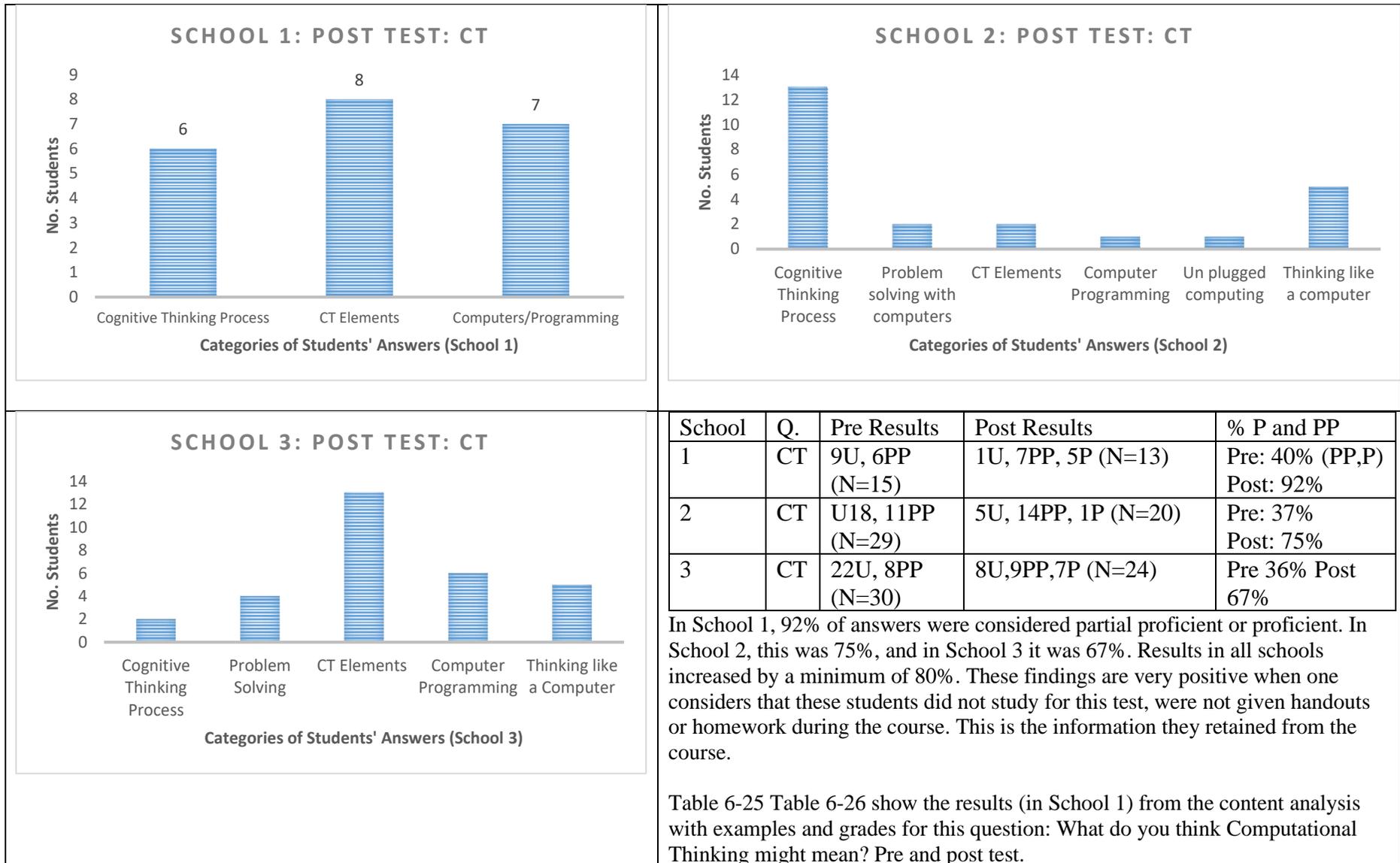


Table 6-25 Results from the analysis of the open-ended question (Pre-Test) What do you think Computational Thinking might mean? (N=15, School 1)

Category and Description	Example of Responses	%
Problem decomposition Answers identify CT as a process that involves breaking down a task into smaller steps	‘the process of breaking down a large task or a large goal into smaller tasks or steps which can easily be completed’ (PP) ‘Things are broken down’(PP)	16.7%
Cognitive Thinking Process Answers mention a specific thinking /cognitive process	‘applying Logic to any task’(PP) ‘also might be logical thinking’(PP)	11.1%
Problem solving using a computer Answers identifying CT as a problem solving technique that is used in combination with computers. Thinking with the help of a computer.	‘Using the aid of computers to help solve day to day problems, Basically solving problems with the aid of technology’ e.g. ‘planning to build a city’(U)	11.1%
Computers/CS and Programming Answers strongly connect CT with programming skills, Computer Science or how a computer works	‘Being able to program/write code’(U) ‘Thinking in coding’(U) ‘Solving problems using maths or computer science’(PP) ‘Thinking about how a computer works and what they can do’(U) ‘learning how a computer works’(U)	44.4%
Thinking like a computer Answers imply that a computer is a thinking body	‘thinking like a computer’(U) ‘computer is making certain decisions’(U) ‘thinking like a computer to get a better understanding of it’(U)	16.7%

Table 6-26 Results from the analysis of the open-ended question (Post-Test) What do you think Computational Thinking might mean? (N=13, School 1)

Category and Description	Examples of Responses	%
Cognitive Thinking Process Answers mention a specific thinking/cognitive process for example logic	‘Computational thinking is a way to break down problems and solve them in a logical way so that a computer can understand it easily’ (P) ‘Computational Thinking is processing problems and logic in the way that a computer would. This involves breaking down a problem, simplifying it and forming an algorithm to solve it’(P)	27%
Answers identify computational thinking elements	‘Use of patterns, logic, algorithms and abstraction to break down large tasks into small tasks, able to be completed by a computer’(P) ‘Computational Thinking is breaking down all the factors to make it simpler on a computer, as a computer has no knowledge’ (PP)	36.3%
Computers/CS and Programming Answers connect CT with programming skills, Computer Science or how a computer works	‘exact orders and algorithms used for programming computers’ (PP) ‘Computational Thinking is how a computer programmes are made. Such as algorithms. It also helps to solve a problem’ (PP) ‘Using the aid of computers and technology on own day-to-day lives.’ (U)	36.3%

6.6.1.4.3.2 Explain what you understand by algorithms? Analysis

The following section displays six bar charts. These depict the categories and frequency per school of answers for Q3: Explain what you understand by algorithms? The categories were developed from students' answers per school. The charts are displayed for each school, pre and post-test. This is followed by Table 6-29, Table 6-30 which show the results from the content analysis of School 3 with students' examples and grades for the question:

Table 6-27 Bar charts depicting the category of answers and their frequency for Algorithm Question pre test.

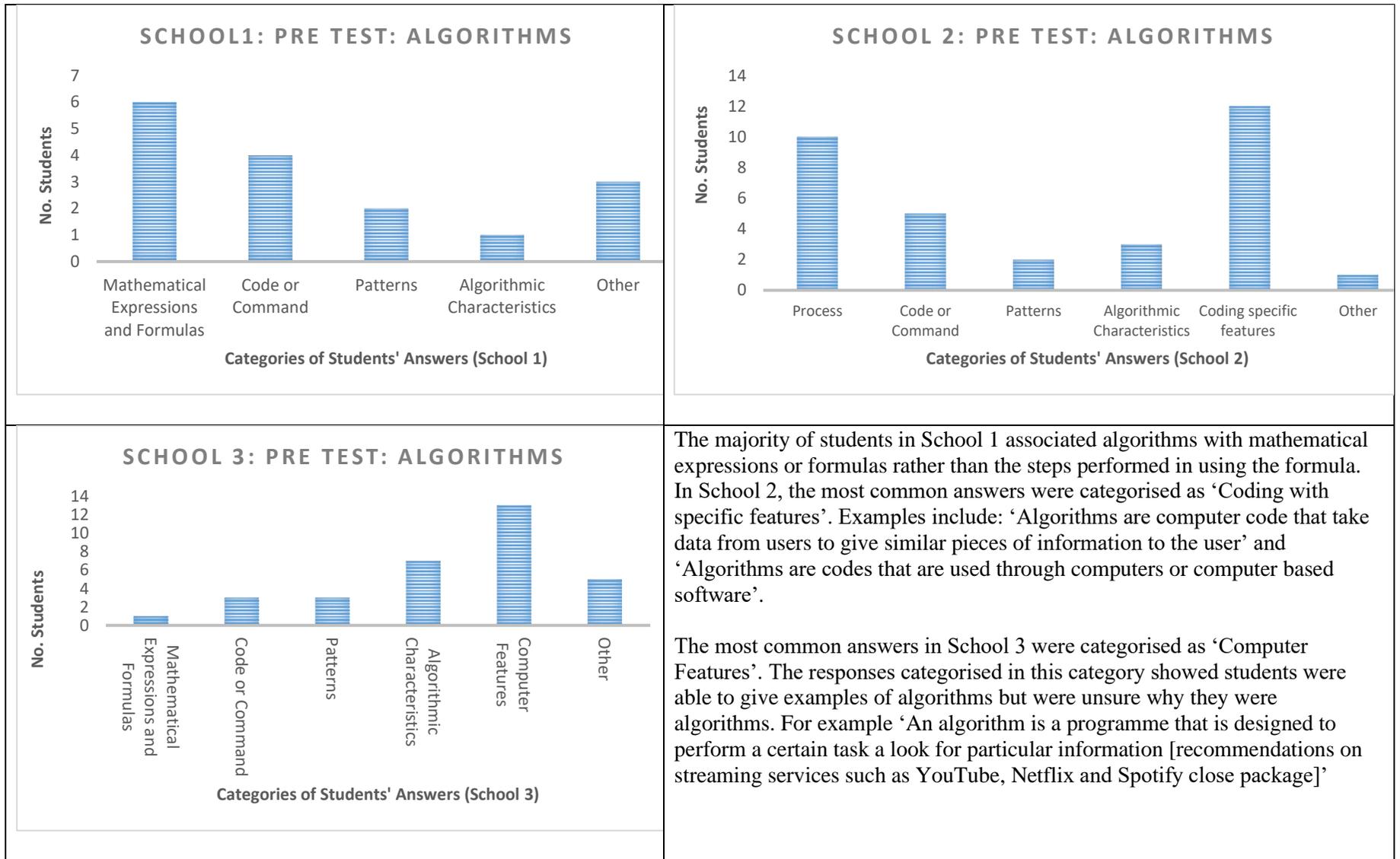
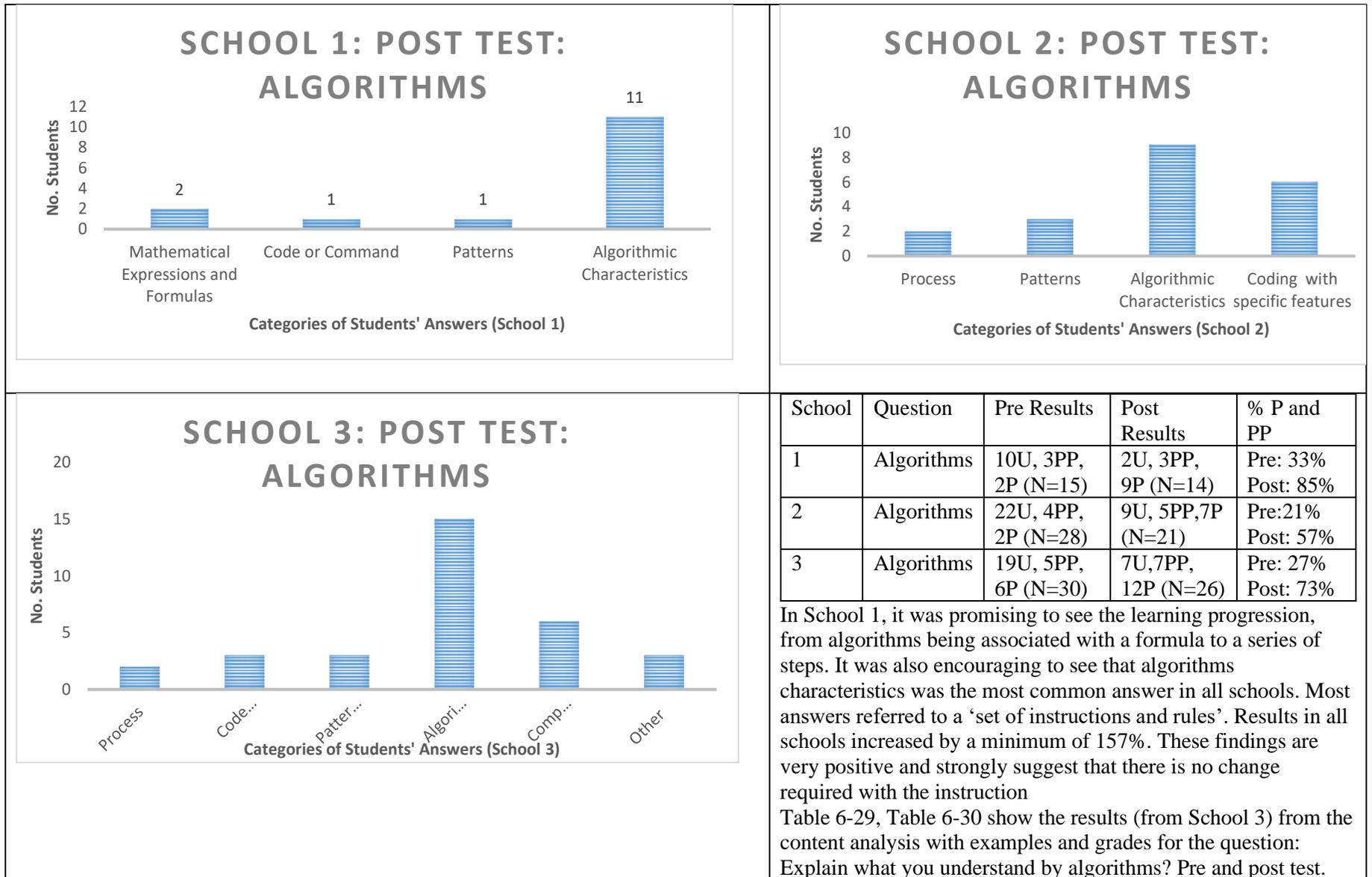


Table 6-28 Bar charts depicting the category of answers and their frequency for Algorithm Question post test.



<i>Table 6-29 Results from the analysis of the open-ended question (Pre-Test) Explain what you understand by algorithms? (N=30, School 3)</i>			<i>Table 6-30 Results from the analysis of the open-ended question (Post-Test) Explain what you understand by algorithms? (N=26, School 3)</i>		
Category and Description	Example of Responses	%	Category and Description	Example of Responses	%
Mathematical Expressions and Formulas Answers strongly connect Algorithms with equations, theorems, statistics and mathematical formulas	‘I have heard of algorithms before however I don’t know what they are . I think they are maths problems or patterns. I also believe it could be statistics’(U)	3.1%	Process Answers strongly connect Algorithms with equations, theorems, statistics and mathematical formulas	the process of how things are made to solve, input and output of information [google, chatbot, alexa, googlehome] (PP)	6.2%
Code or Command Answers strongly connect Algorithms with codes or commands	‘I think they are commands that computers can be trained to do to perform actions or complete tasks’(U) ‘The code that you input something into to figure out the answer or if something is input in something what happened as a result’(U)	9.4%	Code or Command Answers strongly connect Algorithms with codes or commands	Computer code that is used to problem solve the computer and give us an answer or result [Instagram Google Translate predictive text](PP) Algorithms are computer programmes (U)	9.4%
Patterns Answers connect algorithms to being patterns	‘Algorithms are certain patterns that are based off of collected data. For examples Instagram sets algorithms as to what posts you see based off of what posts you like most views more frequently’(PP) ‘Consistent patterns , they can help solve problems by analysing behaviour and reactions, and outcomes.’(U)	9.4%	Patterns Answers connect algorithms to being patterns	I understand that an algorithm is a pattern of instructions. I think they are shown in programmes such as autocorrect and spellcheck (P) Algorithms are patterns or sets of rules used to complete tasks and solve problems. They are consistent. If performed correctly an algorithm will work every time.(P)	9.4%
Algorithmic Characteristics	‘I think algorithms are a set of rules that have to be followed using a	21.9%	Algorithmic Characteristics	‘A set of rules/steps that you follow without questioning too much (recipes	46.9%

Answer mentions characteristics of algorithms such as logical ordered steps performed in sequence to solve a problem	computer (EG rule in school we must follow, rules making lego, rules must be followed correctly so that the finished project works out)'(P) 'Algorithms are a set of rules to be followed in problem solving operations' (P) 'set of rules to be followed in problem solving'(P)		Answer mentions characteristics of algorithms such as logical ordered steps performed in sequence to solve a problem	for cooking, lego, instructions for making something)' (P) 'I understand algorithms to be a set of rules that are blindly followed by a computer in order to give a specific output (computer program that draws a smiley face)'(p) 'Algorithms are a set of rules that you blindly follow (card tricks, Instagram, social media)'(P) 'Algorithms are lists of commands or steps/ a sequence that is used to figure something out or to do something [winning X's and O's with a strategy or sequence of moves / chess '(P)	
Computer Features Answers highlights how student has heard the work in reference to data or website or computers. The student can provide examples but does not understand what the word means	'Algorithms are certain patterns that are based off of collected data. For examples Instagram sets algorithms as to what posts you see based off of what posts you like most views more frequently'(U) 'I don't know, is it to do with data that is collected' (U) Collecting data off a computer (a survey) (U)	40.6%	Computer Features Answers highlights how student has heard the work in reference to data or website or computers. The student can provide examples but does not understand what the word means	A program that has a certain way of collecting data (spellcheck, google search, chatbots) (U)	18.7%
Other Unique individual answers that did not relate to the other categories. unknown	Answers empty or 'I don't know' (U)	15.6%			
			Other Unique individual	algorithms is a number of ways to do something like crack a password. [card tricks, Instagram, social media] (U)	9.4

	<p>answers that did not relate to the other categories.</p> <p>Mixed word up with abstraction, process</p>		
--	---	--	--

6.6.1.4.3.3 Ethical and Logical Results

Students were also asked the following two questions (post test only):

Do ethics have a place in Computational Thinking? Please provide an example to explain your reasoning. Does Logic have a place in Computational Thinking? Please give an example to explain your reasoning. Full results from the analysis of these questions are displayed in Appendix K. In summary, most students agreed that logic was important to Computational Thinking, correctly citing validity for their rationale. Similarly, most students agreed the ethics is important in Computational Thinking. Expert 1 highlighted the importance of exemplification. This is particularly true of the Ethics part of the course. I need to provide more examples of the importance of ethical thinking in connection to writing algorithms. All examples provided by students were the driverless car.

Recommended changes for the course from the analysis of these questions were as follows:

- include in Version 2 of the teacher's guide, information on the German code of Ethics for programmers connected to 'Automated and connected driving'
- ensure students know what the word ethics means
- change the question, 'Does Logic have a place in Computational Thinking?' to 'Does Logical Thinking have a place in Computational Thinking?' As it will remove confusion with Boolean logic and Logic gates.

6.6.2 Engagement (Teachers' Perspective)

This section concerns engagement. This topic had previously been discussed in section 6.5.1.1.1. where students' quantitative results showed that they found the course engaging.

The section concerns teachers' perspective on engagement. It was the third theme captured in the Thematic Analysis of teachers' qualitative data.

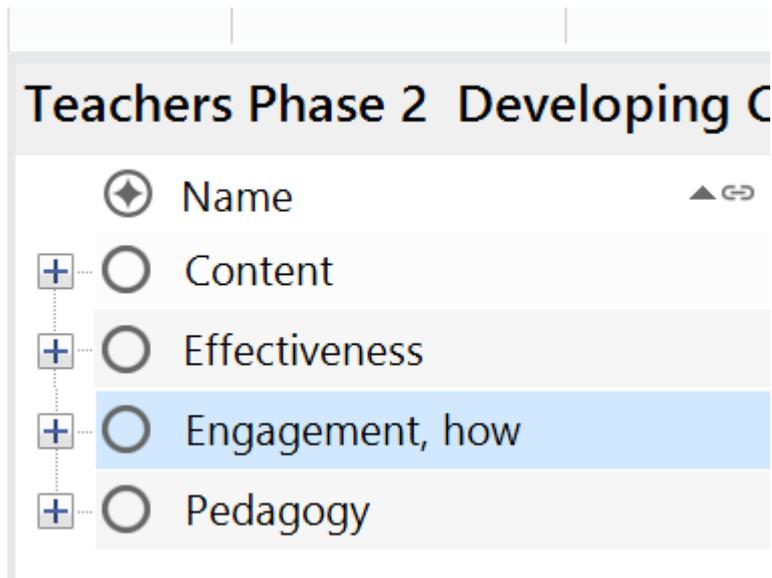


Figure 6-31 NVivo Screenshot of themes related to the teachers' perspective (V2)

It contains four sub-themes: 'Validation', 'Interest,' 'Participation', and 'Pedagogy'. The 'Validation' theme is concerned with validating the findings from the Student Engagement questionnaires and also the 'On the Fly' analysis. The other three sub-themes were created inductively to capture teachers' views on what made the course engaging. These findings informed the instructional design of the course.

6.6.2.1 Validation

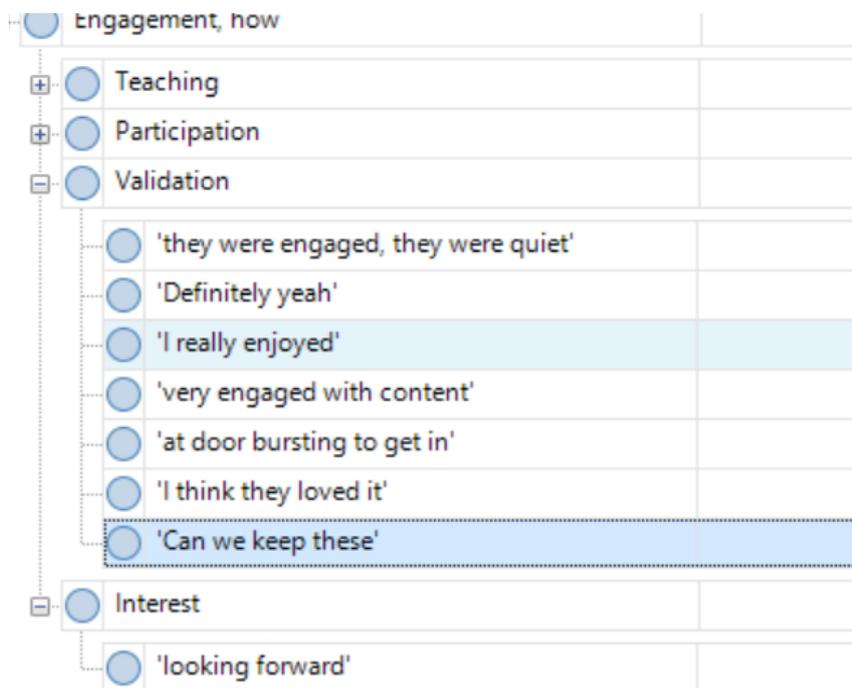


Figure 6-32 NVivo capture of the Validation codes for Engagement

The 'Validation' subtheme confirmed that teachers found the course engaging and believed that their students did also. Figure 6-32 displays the positive codes captured under this subtheme. Two quotes from Teacher 2 and Teacher 4 confirm same.

...and I know you were very good at asking, 'Look is everyone with me?' I mean they were and you know sometimes you can tell if lads are spoofing, if they're just saying, 'Yeah', to move on but they genuinely you could see there was – they were engaged they were quiet, they were actually understanding what you were explaining yeah. (Teacher 2)

At the start when I said we are doing Computational Thinking. They didn't even know what it was. Now, every week, they are at the door bursting to get in.' (Teacher 4)

6.6.2.2 Interest

This subtheme examines the perceived relationship between engagement and interest documented by teachers. The word interest is understood to mean both students' interest in and how interesting (or not) they found the course. Teacher 1b highlighted the relationship:

I found it really interesting all this stuff that you were teaching and I think the boys engaged on the same level, most of them, as me, so yeah I would think it would be like I can't think of a week that I didn't enjoy sitting there listening you know that kind of way, so I think it was really engaging, yeah (Teacher 1b)

The converse of this relationship was also found to be true. Of the thirty students who participated in the course in School 3, four to five students were observed to not be interested in the course. The teacher experienced the same lack of engagement with the students in a different class.

It was disappointing that they didn't engage ... and I've also been with them in business where they were kind of much more laid back and letting people do their job. (Teacher 3)

The unplugged activities were also discussed in relation to engagement. Teachers' reported that students found them interesting which influenced engagement. The magic tricks were specifically mentioned.

Yeah I don't think me or them understood how that actually worked. I don't think any of us knew where that was coming from but I think it was interesting for them to – it kind of blew their mind a little bit yeah and they blindly followed yeah exactly. Just followed instructions isn't it. (Teacher 1b)

That one was good - they were all a bit apprehensive. I don't know what the word is, but they were kind of like distrusting of you as to how is she doing this? Then they were all watching. Even the ones who were like just too cool for school; their interest is peaked, and they were kind of looking in. (Teacher 4)

Students' interest was also linked to the interactive and collaborative nature of the activities.

Me: Okay so I guess so the first series of questions I'm going to ask is related to ... the content and whether you thought it was appropriate for Transition Year students in an Irish context.

Teacher 1b: I think it was, yeah, I found even looking at the boys their interaction with the content that you were doing was really, really positive, you could tell that like they obviously, it was something that they were interested in and maybe

something that they don't explore in school in another class or module or anything like that, it doesn't exist at the moment really in post-primary schools but you could tell that the boys had interest (Teacher 1b)

6.6.2.3 *Participation*

Student engagement was also linked by teachers to participation. The unplugged activities allowed for everyone to get involved. They were reported as being pitched at the right level (Teacher 2). Teacher 1a concurred with Teacher 2, reporting that the concepts, pictures and apps used in the course were everyday objects that students were familiar with. They allowed everyone to be involved.

Even the weaker student that was there was able to mix and match buttons. So, that gave them a little bit of confidence. Then, you know, different students at different levels would have got the concept. Everybody was able to do something no matter what cognitive development they were at. No one felt they couldn't write anything down or say anything. I thought it went very much from easy to more difficult, more difficult and then you brought it back and summarised it all together. So, obviously, students got it at different levels but for the most part everybody was engaged. (Teacher 1a)

The converse of participation was also reported. Teacher 1a reported the lack of student activity in the first lesson as an issue. The first lesson piloted of this course had only two puzzles and one activity near the end of the 80 minutes class. This setup was changed for future classes

There wasn't too many questions being asked of them. The fact that they actually couldn't do anything for a long periods during the lesson they were passive, and they could tune out. I felt that very few of them could actually engage with you. (Teacher 1a)

6.6.2.4 *Pedagogy*

Pedagogy is the last sub-theme captured under the engagement category. This subtheme links engagement with 1) unplugged activities, specifically their competitive aspect and their hooks and 2) instructional clarity.

<input checked="" type="radio"/> Pedagogy	0	0
<input type="radio"/> No instruction	1	2
<input type="radio"/> Real World Examples	2	2
<input type="radio"/> Exemplification	2	4
<input type="radio"/> Groups Competiveness	3	7
<input checked="" type="radio"/> Engaging Activities	3	4
<input type="radio"/> 'references' 'stand out'	1	1
<input type="radio"/> 'Fun'	1	2
<input type="radio"/> Clarity	4	8

Figure 6-33 NVivo screenshot of sub-theme Pedagogy

Two teachers specifically linked the competitive nature of the unplugged activities to engagement.

Because they were working in groups, they were kind of competing against each other which helped bring them in. Anyone who wasn't engaged it helped bring them in. They wanted their group to win. So, everyone did get involved (Teacher 4)

Students were at their most engaged when there was a competitive aspect such as the chilli and chocolate game/ card trick. (Teacher 2)

The lead-in and hook to activities were also mentioned as a key instructional technique that encouraged engagement.

I think they loved it especially if you did an example first or if you showed them something and they had to figure out how it was done, either the code breaking thing or any of the games and activities, they were engaged. (Teacher 4)

This demonstration or hook before an activity aided clarity, which teachers voiced as being important to engagement. It also showed students what success was.

And then there was some aspects I think was it the – maybe it was the spell checker and there was one other, there was a couple of things there like you said where it's just maybe just a little bit more scaffolding before it so that they were very clear and concise about what the task was at hand, you know that there was no confusion because I think that was – that's what decreases motivation a lot as well (Teacher 2)

Students were also most engaged when the task at hand was very clear to them. (Teacher 2)

Isn't that the whole notion of success criteria. You know, we tell them you can do this, this and this and if you do this, you will achieve this. (Teacher 1a)

6.6.2.5 Engagement Summary

In Section 6.6.1.1.2, students stated that the activities were a strength of the course. They found them interesting, engaging, fun. In this section, teachers identified characteristics of said activities which they believe impact student engagement. The characteristics are, competitiveness, clarity, lead-in hooks, interesting content, interest in course and accessibility.

6.6.3 Content

Name	Files	References
Engagement, how	0	0
Content	1	1
Reactions	0	0
Assessment	0	0
Characteristics	0	0
Activities	1	1
Positive	4	26
Corrective Feedback	6	25
Pedagogy	1	1

Figure 6-34 NVivo Screenshot of Content Theme

The third theme captured from teachers' qualitative data was Content. In particular, it captured teachers' 1) emotional reactions, 2) views on assessment and activities, and 3) course characteristics. Elements of this theme: 'Teachers' Reactions', 'Activities', and course 'Characteristics' overlapped with previous themes and thus are not discussed here.

6.6.3.1 *Assessment*

Teachers were asked their views on recommended assessment strategies for this course.

They put forward the following ideas, whiteboards, exit cards and the KWL (know, want to know, and ultimately learn) strategy. They collectively advocated an approach that was simple, short and quick. Teacher 3 advocated whiteboards as a way of recording ideas, and answers in the classroom. Students could then take pictures of their answers and load to the Google Classroom. Teacher 2 and Teacher 1b advocated exit cards as an anonymous and simple way of capturing student learning at the end of the class. This researcher noticed that during the class, especially the ethics lesson, students were reluctant to write their answers, they wanted to say them instead.

ME: I noticed one of the ways even with doing the placemats, sometimes they wanted to tell me rather than write it.

Teacher 1b: Right okay. Yeah I remember a few times you were saying, write it down

Teachers' ideas were taken on board in relation to the assessment for phase 2 (see 7.1.3.2.1)

6.6.4 **Pedagogy**

The fourth theme captured was Pedagogy. In particular, it captured characteristics and recommendations. Elements of this theme do overlap with other themes see 6.6.2.4.

6.6.4.1 Characteristics

Characteristics	0	0
Active	0	0
'doing'	1	1
enhancing practical	1	1
Explanations	1	1
Teaching Programming five fund	1	1
Reflection	1	1
Scaffolding	1	1
'process involved'	1	1
Exemplification	2	2
Setting the scene	2	2
Competitiveness	3	7
Discussion	3	3
Groups	4	6
repetition or recap	4	6

Figure 6-35 NVivo nodes illustrating the codes captured under the Characteristics category

Teachers highlighted many positive characteristics related to the pedagogy see Figure 6-35.

They can be listed as follows, collaboration, discussion, reflection, exemplification, repetition, recap, linkage scaffolding and competitiveness. These attributes have previously been discussed and thus are not reviewed again in this section. However, a quote that stands out is in relation to reflection and the topic of abstraction, which was taught using the 30 seconds game

The specific questions were great to get them to think about how they solve problems especially playing '30 seconds' after they had done the riddles. Asking them how the guessed correctly was a great addition, they rarely get to think about that kind of thing. (Teacher 4)

6.6.4.2 *Recommendations*

There was only one specific recommendation given at the interview stage which was not captured during the ‘On the Fly’ analysis, and that was a recommendation by Teacher 3 that I should give homework to the students.

6.7 **Chapter Summary**

This chapter presented a detailed narrative of the revisions and evaluation of Version 1.

The many revisions that occurred during the ‘On the Fly’ stage were documented and the rationale provided for their inclusion. These revisions were further explored to advance the development of the course’s instructional framework. Of note, a pre-activation step was introduced to establish a baseline of Computational Thinking knowledge. Clarity was validated as being an important instructional element, as it was linked to both the quality of the course and student engagement. The use of videos as a lead in hook for activities, to demonstrate concepts, to aid understanding, to stimulate the imagination and to show that some unplugged activities were based on real-life events was invaluable. Knowledge being demonstrated first to students before application, was shown to aid clarity and engagement. Teachers identified activities specifically the following characteristics: competitiveness, clarity, lead-in hooks, interest and accessibility as having an impact on engagement. Key components of the evolving instructional framework were identified in this phase: unplugged activities, demonstrations, application, pre-activation, clarity and reflection.

The course was evaluated to ascertain if it was of high quality, engaging and effective.

With reference to quality, two content experts validated the course for content validity.

They confirmed that the course included relevant subjects and topics, and that it had the potential for developing students’ understanding of Computational Thinking. Both teachers and students found the course to be engaging. Students stated that the ‘Activities’, ‘Pedagogy’, ‘Learning’ and ‘Course Content’ were particular strengths of the course.

Students were engaged with both the content and activities. These findings suggest that the

course content is age-appropriate and relevant for students. The pre-tests also confirmed same as they showed that most students did not have previous knowledge of the content, and thus would not be bored. Students identified Computational Thinking, algorithms, problem-solving, coding and ethics as topics that they had learned. All teachers agreed that they felt confident teaching the course. The finding from the post tests confirmed students' learning with regards to Computational Thinking and algorithms.

Findings from Version 1 were incorporated into Version 2. Chapter 7 is concerned with the piloting of Version 2.

7 Prototype Phase Version 2

This chapter concerns the prototype phase of this study, specifically the prototyping of Version 2 of the Computational Thinking course, including the teacher workshop. This phase was conducted over four months, from the end of August 2019 to December 2019. It involved three hundred and forty students, six teachers, and two schools. The goal of this version of the prototype is to explore the practicality and usability of the course. Data is gathered using various means: individual and group interviews with teachers, teacher questionnaires, teacher diaries, student artefacts and student multiple-choice questionnaires. The analytic approach is mixed; it involves thematic analysis as well as descriptive statistics. The collected data forms part of the solution in that it contributes to the instructional design but also aims at answering the following questions (which inform the research question):

- Practicality: Does the course work in a school setting? Is the content (teacher guide and course materials) usable and appropriate for the teachers?
- Effective: Is the course effective?
- Low threshold: Is the course low-threshold?

This chapter presents the experience of the six teachers who taught this course. Student assessment is also investigated with respect to student learning and instructional design. The chapter starts by describing the two schools involved in this phase and their respective teachers. The purpose of the descriptions is to aid in contextualising the findings from Version 2. This is followed by a discussion and findings from the workshop conducted before the prototyping of Version 2. The findings from the teachers' experience teaching the course is next presented. These findings help to answer the above questions.

7.1 The Setting

Two schools were involved in the prototyping of Version 2 of the course, School 2 and School 5. School 5 was new to the prototyping phase. This school had responded during the initial recruitment drive in October 2018 to state that they would be involved in this phase. Due to timetabling constraints, Schools 1, 3 and 4 (from pilot 1) were not part of this prototype phase.

7.1.1 School Details

School 2 is an all-boys school outside of Dublin. The participating teacher self-selected into the study and was interested in teaching Leaving Certificate Computer Science in the future. He has a Higher Diploma in Software Development. He was an observer in the pilot of Version 1. In this pilot, he taught the course to two Transition Year class forms, twice a week. The lesson length was forty minutes and was taught during a timetable slot for computers. Four visits were made to this school during the study period.

School 5 is an all-boys school in the south of Ireland. The course was taught by five teachers, whom all attended the workshop (see 7.2), three females and two males. Three teachers had previous computer/programming experience, graduate diploma or part of a minor degree, and the other two teachers described themselves as having very little to no previous computing/programming experience. The course was taught to thirteen class forms, twice a week. The classes consisted of four first-year class forms, four second-year class forms, four third-year class forms and one Transition Year form. The lesson length was forty minutes and was taught during a timetable slot for computers. Five visits were made to this school during the study period.

7.2 Workshop

The workshop took place in School 5, on the 28th of August 2019. Its purpose was to teach the course to the participating teachers (as they had not previously been course observers).

It took a full day and was delivered the exact way it was taught to the second-level students in the prototyping of Version 1.

7.2.1 Workshop Findings

A questionnaire was issued at the end of the workshop (see 4.5.2.3). Its purpose was to evaluate the workshop and capture the new teachers' perspectives on the course's quality, content, practicality, and confidence in teaching said course. It contained nineteen Likert items and three open-ended statements. Appendix N contains the results from the Likert Statements. The answers to the open-ended statements were coded using both deductive and inductive thematic analysis. The initial categories were created to reflect the open-ended statements (deductive *a priori*): 'First Years', 'Meets Needs', and 'Teachers (low threshold)'. The sub-themes emerged from the data using inductive coding (Fereday and Muir-Cochrane, 2006). A hierarchy chart of the codes that resulted from these three statements is provided below see Figure 7-1.

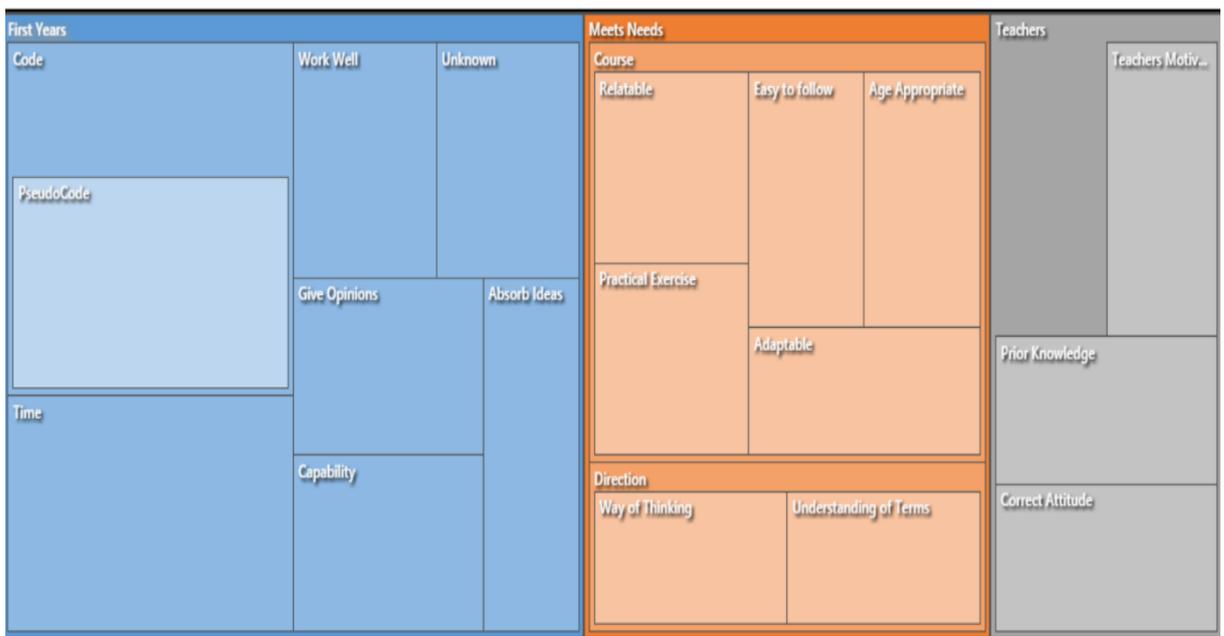


Figure 7-1 An NVivo Hierarchy chart generated from the workshop open-ended questions

7.2.2 Effectiveness

The quantitative and qualitative data were analysed to ascertain how effective both the workshop and the course were. This was achieved by ascertaining the teacher's reactions to, and learning from, the course (Guskey, 2002). From Appendix N, it can be seen that the median and mode value for every Likert item except one was 5.

7.2.2.1 Effectiveness: 'Meets Needs'

All teachers strongly agreed that the course met their needs. The median and mode value for this item was 5. Whilst one teacher's answer to the Likert statement was missing (see Appendix N), they answered the related open-ended statement, 'The course meets your needs' with: 'Far exceeded expectations – found the practical exercises really good and age appropriate + made it very relatable' (Teacher 7). Teacher reactions to the course were overwhelmingly positive, a sample of captured codes is as follows: 'relatable', 'practical', 'adaptable', 'easy to follow'.

7.2.2.2 Participants Learning

All teachers professed agreement that: 1) they were confident to teach the course, 2) they understood the course content and design, and 3) that the course materials were easy to use (modes for all items were 5 see Appendix N). All teachers indicated that the workshop was informative and provided them with new information. The quantitative results suggest that teachers acquire the intended knowledge and skills to teach the course.

The day after the workshop I received an email from Teacher 7. This thesis opens with passages from this email (see Chapter 1), where the teacher stated that her approach to teaching Computational Thinking and her knowledge of same fundamentally changed after attending the workshop.

7.2.3 Low threshold

Data was also captured in relation to the ‘low threshold’ characteristic of the course. Once again this information was captured using both qualitative and quantitative means.

All teachers agreed with the statement that the course met the goal of assuming no previous Computer Science knowledge is necessary for students. The statement that the course content requires no previous Computer Science knowledge for teachers provided a more striking observation. Three teachers strongly agreed with the statement, one teacher disagreed, and one was neutral. Surprisingly of the three teachers that strongly agreed, two had no previous Computer Science experience. One teacher stated ‘Absolutely yes’ with the another commenting that ‘As a teacher with little classroom computer science experience this was easy to follow and useful’. The two teachers who did not strongly agree, both had Computer Science experience. They suggested the following: ‘I think teachers will need prior knowledge to have the confidence to teach the subject’ and ‘I’m not sure – the main thing I think is a willingness and openness to the subject – once teacher has correct attitude I think any teacher could relate to the course’.

7.2.4 Course Content

The teachers at school 5 needed a Computational Thinking course suitable for first year students. As the course was initially designed for Transition Year students, I was cognisant that the course might not be suitable, (although I had simplified theoretical elements of the course in preparation for this).

Teachers were also asked to respond both quantitatively and qualitatively to the following statement ‘The course content is appropriate for first-year students’.

The mode value for this question was 4 (which was the only item that did not receive a 5 value). The open-ended statement revealed that teachers felt that most of the course was

appropriate for first years except for week 5, where three teachers were concerned that the coding part was too difficult.

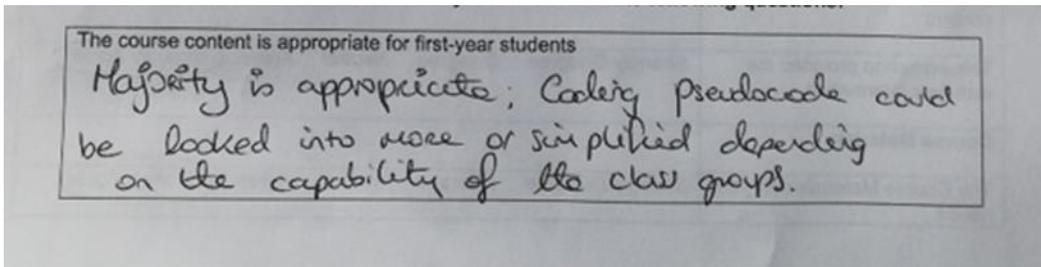


Figure 7-2 Snippet from a teacher's answer.

After the workshop, the teachers decided to teach the course to thirteen class forms ranging from first year to Transition Year students.

7.2.5 Changes to content

Two changes to course content resulted following the workshop. They were as follows. A new screencast was developed to explain the SpellChecker code as the previous screencast had no audio. The pseudocode examples in Week 5 were also simplified (the min and max and average activity were removed, the Did I Pass activity was made into two parts).

7.3 Finding from Version 2: Teachers' perspectives

The following section presents the teachers' perspectives from the piloting of Version 2 in two schools. Data was collected using the following methods: individual and group interviews with teachers (five teachers), teacher questionnaires (workshop) and teacher diaries (all six teachers). The data was analysed using thematic analysis. The findings contributed to the course's instructional design and also an evaluation of its quality, specifically its practicality, sustainability, and effectiveness (teachers' reactions, teachers' use of new knowledge and skills, teachers' learning, and student learning outcomes).

The data from all six teachers were analysed collectively using thematic analysis. Four key themes were generated: 'Teachers' Experiences', 'Students' Experiences', 'Learning

Environment’, and ‘Course Design’. The following section provides a discussion of these captured themes.

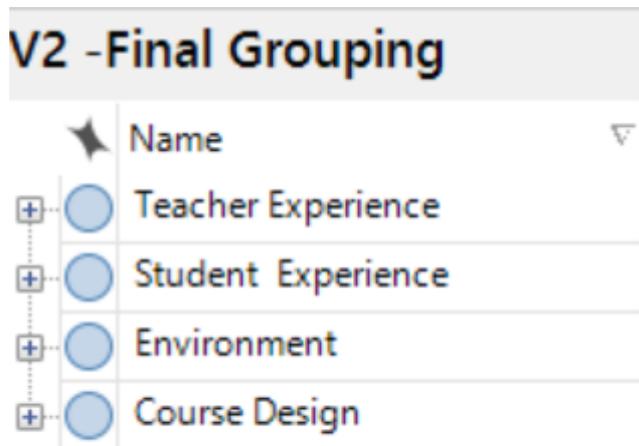


Figure 7-3 An Nvivo Screenshot displaying the four themes generated from the teacher qualitative data

7.3.1 Theme 1: Teachers’ Experience

This theme documents the teachers’ experience with teaching the Computational Thinking course. This theme has four sub-themes; ‘Teachers’ Learning’, ‘Classroom Teaching’, ‘Reactions’, and ‘Readiness and Sustainability. The sub-themes of ‘Reactions’ and ‘Teacher Learning’ were created deductively based on Guskey’s model (deductive *a priori*). The following section explores each of the sub-themes in detail

7.3.1.1 ‘Teachers’ Learning’

This sub-theme is concerned with evaluation, specifically the effectiveness of the course, and how this connected to teachers’ application of their new knowledge and skills? (Guskey, 2002). Four teachers in school 5, previously taught Computational Thinking. Their second-year students received: two classes on Computational Thinking, one class on algorithms, and two classes on flowcharts and pseudocode (Teacher 10). Regarding their previous teaching of Computational Thinking, Teacher 10 outlined that his main problem with the content was a lack of knowledge. He outlined how he had no training in this area and would use explanation videos to teach the concepts: ‘you don’t have that real-world

example to hand, and you're throwing on a YouTube video' (Teacher 10). Teachers 6 and 7 concurred with this assessment stating that they were just going through the motions last year: 'I felt I was just putting up slides last year. Getting through it.' (Teacher 7).

All six teachers concurred that they now had the required knowledge and skills to teach the course. Four teachers specifically stated that there was no comparison with what they knew 'this year' compared to last year. Previously, they had heard of abstraction and decomposition, but now they have clarity, confidence, and understanding associated with those words. What was interesting was that they now saw the problem-solving element of Computational Thinking as being relevant outside of Computer Science

problem solving, like even the four aspects; decomposition, abstraction etc., that related to every problem they'll come across in life not just computational thinking. So, I try and relate everything they're doing to life as well as obviously computational thinking. And it's just so relatable, it really is, it's so relatable. (Teacher 8)

Together with their students, teachers related Computational Thinking to other subjects such as Art, Maths, CSPE, Religion, Business Studies, and English. This was reasoned as being due to the course's relatability and their now new understanding of said content, resulting in better articulation of Computational Thinking concepts. Teacher 6 stated that she used the word abstraction when doing modelling in trigonometry. Teacher 8 alluded to how the AI and ethics content is cross-curricular to subjects such as CSPE and religion. Teacher 10 stated how he linked logic to English writing to check if conclusions are valid. Teacher 9 explained how her students stated they would use the idea of decomposition and abstraction in their Art class when drawing an image. Teachers 6 and 7 explored this topic further by stating that the course had given them an articulation grammar

[Teacher 7] And particularly giving me the language to use the word abstraction.

[Teacher 6] Abstraction and decomposition.

[Teacher 7] Like before, as I said, we did a bit of it last year, I never used the words outside of those two classes. But now I can see myself using it for lots of different classes, like maths

[Teacher 6] Yeah, in Maths.

In both their interviews and emails, teachers' demonstrated their knowledge of Computational Thinking frequently. They taught the course content using the proposed instructional design, but where necessary, adapted and modified the content to better suit their class. An example of teachers' use of this new knowledge and skills is clearly demonstrated by Teacher 8 (who had no previous knowledge of Computer Science and Computational Thinking).

And you're abstracting the stuff that is important and you are decomposing the problem of 'I want to go to the cinema'. The problem is what are you going to see, when, and how? How, where, and stuff like that. So, that was the decomposition of that. Not a problem but it was an issue and it was one that they all relate to. (Teacher 8)

He changed the course example from a Dublin Bus app to that of a cinema app to better illustrate decomposition and abstraction. His explanation clearly shows his understanding of the concept.

7.3.1.2 Readiness and Sustainability

Readiness and Sustainability were two further subthemes captured under the 'Teachers' Experience' theme. These sub-themes were concerned with teachers' readiness to teach the course, and with the intervention's long-term sustainability. After the workshop, teachers were asked if they felt confident teaching the course; they all responded affirmingly. This question was repeated during their interviews. All teachers re-iterated that they felt prepared to teach the course and stressed that the workshop was essential to this preparation. 'I wouldn't have been able to teach it probably without the workshop' (Teacher 6). There was one negative comment concerning readiness. Teacher 8 (who had no previous Computer Science experience) found week five difficult, in that 'even after teaching it to three classes, I wasn't 100% confident'. This is an interesting observation as

it highlights how the programming element affected the low threshold characteristic. Week 5 was the only week that had its sole focus on the subject domain of Computer Science.

Teacher 2, who was an observer in the pilot of Version 1, outlined how his ‘biggest obstacle’ to teaching the course, was his ‘unfamiliarity with content’. However, he recognises that his teaching of the same content to his second TY class form ‘always runs much smoother due to my confidence boosted having done it once.’

All teachers agreed that they are going to teach the course again. However, for School 5 they recommended the following change, concerning topic sequencing. They would divide the course into two time blocks, where they would teach Unit 1, Unit 2 and Unit 5 together (time block 1) and then start teaching programming. They would then teach Unit 3 and Unit 4 at a later time (time block 2).

Oh, absolutely, yeah, because there’s a great framework around what we’re doing and why we’re doing it so other than probably shortening aspects of it, I would certainly do it again. (Teacher 7)

Teachers 6, 7, and 9 highlighted how they strongly believed that their students retained zero to little of the content last year when they taught Computational Thinking. However, what makes this course more teachable and relatable to students, (and thus sustainable) is the fact that the unplugged activities are memorable. As teachers, they now have a series of activities that they can relate back to. Teacher 9 articulated that as the course progressed, she enjoyed relating back to specific unplugged activities and hearing the students articulate Computational Thinking terminology associated with the activities.

being able to relate back to the 30 second game and relating back to it and they’re going, yeah, and they’re able to come up with the word then and the correct terminology and everything, which I was fascinated by. I enjoyed getting it back from them whereas we’d never had that before. Just kind of gone over their head. (Teacher 9)

Teacher 6 stated that when they start programming in January 2020, decomposition and abstraction will be introduced with its reference to categorising buttons. Curzon (2014b) argues that unplugged activities (and storytelling) allow for Computational Thinking to be explained and taught in powerful ways. In a follow-up study, Curzon *et al.* (2019) confirm how these activities are powerfully memorable but they are cognisant that much work on unplugged approaches is anecdotal. In this doctoral study, teachers confirmed that the activities provided significant reference points. Like the pre-activities that lay a baseline to explain Computational Thinking concepts, the pre and unplugged activities were going to be used as concrete reference points when programming.

7.3.1.3 Teaching

Three key points resulted from the sub-theme of teaching: changes to teachers' teaching style, the need for written reflections/assessments, and classroom management issues. Each of these changes is discussed in detail in the following section.

7.3.1.3.1 Writing

Most teachers confirmed the prominence of the copybook and writing in their everyday teaching style: 'copy work, backed up with theory, backed up with copy work' (Teacher 9). Writing was described as an important feature of their teaching style, especially for classroom management and ensuring all students' participation. The pre-activation activities such as the button categorising and the 30 seconds game caused classroom management issues for half of the participating teachers. During and after this activity, the students were louder than usual, and the class was harder to control.

Did the 30 sec game again - students were very engaged but as was said by another colleague the problem is bringing them back from this to try and get them back on task - very high energy - a lot of discussion and talking - was really hard as the teacher to bring it back to a level where I could teach without really struggling to get them back on task.' (Teacher 7 journal)

To overcome this, two teachers suggested writing activities.

We're all seeing discipline issues where they are louder than usual and having to be reined in a bit every now and then. I'm overcoming this by getting them writing (always effective to calm a group down!) so every now and then they take some notes into their computer copies, usually to give me a break :) (Teacher 6, email)

Writing was thus suggested as a classroom management technique to calm students. It was also suggested as an alternative to verbal reflections. Reflections formed an integral part of the instructional design of this course. They were incorporated into the design initially due to Merrill's principle for integration. Merrill (2013, p. 20) defines his 'integration principle' as follows 'learning is promoted when learners reflect on, discuss and defend their newly acquired skills'. The course's reflection part was verbal oriented. For some teachers, this contributed to classroom management issues. They reported that not all students engaged with the discussion, and those that did not were disruptive. (The placemat exercise see (Table 6-3) did not work for some class forms see).

The teachers in School 5 particularly highlighted how writing is a core part of their teaching style, influenced by the Irish Examination system. The Irish second level is dominated by two high-stakes written exams, the Junior Cycle and the Leaving Certificate (Smyth, McCoy and Banks, 2019; NCCA, no date). Teacher 8 acknowledged the contradiction in the Irish Educational system where the 'junior cycle is very much active learning but at the end of the junior cycle, you're still a 90% test'. Thus, a change proposed to the Computational Thinking course design highlights that the reflection should take the form that best suits teachers, either that of written or verbal. The rationale was course sustainability, reflection was important for the design, be it verbal or written. In school 5, this was written, as the teachers best felt it would ensure all students participated. The copybook was also stated as forming an essential part of the teaching process. Students use it for reference.

I would have liked at the end of it maybe if they had a computer copy where they just had maybe four or five pages of that computational course that they've just finished. (Teacher 8)

This point has great merit when one considers the students did not have a coursebook. Thus, one of the course's flaws is that the verbal interactions were not backed up with handouts and/or copy of the slides.

7.3.1.3.2 Change in Teaching Style

Two teachers reported mindset changes concerning their teaching style, specifically their relationship with noise. Mindset is understood as a 'person's way of thinking and their opinions' (Cambridge Dictionary, 2021, para. 1).

Both Teacher 6 and Teacher 9 stated they were uncomfortable with noise in the classroom. Reflection on this topic by both teachers resulted in them associating in part their classroom management issues with this discomfort.

[Teacher 6] Maybe I'm the problem, that's what I've taken from this.

[Teacher 9] I have too!

As the course progressed, teacher 9 highlighted the change in her comfort with noise in the classroom.

[Me] So, I'm just going to ask you how you found teaching the course?

[Teacher 9] there was more noise in the classroom and it was trying to get used to that and I remember saying to you at one stage, it's definitely my problem here. The lads are actually interacting with me, there's a nice bit of noise, they're able to discuss topics. And it was just the first couple of weeks getting used to that as opposed to them being on a computer and actually doing things and bringing the noise back into the classroom, and it turned out being really, really positive.

This linkage between noise and classroom management was also made by Teacher 2 and Teacher 8. Teacher 2 and Teacher 8 reported no classroom management issues. They believed this was due to their relaxed teaching style and the fact they were comfortable with noise. Teacher 2 taught practical subjects and highlighted how noise and students talking was a feature of his classes. Teacher 8 highlighted how comfortable he was with noise in the class

there shouldn't be any discipline problems because unless you don't like your students talking or, you're a teacher where they have to listen to me at all times, not allowed... I'm a little bit, you probably saw yourself, a little bit more flexible. A little bit more fluid in teaching and then. When I see them interacting and talking about the subject, that's good for me, that means that they're learning. That means that they're attempting to learn and once you have the material there for them, you're occupying their time and you know, it's just all about active learning at the moment and the junior cycle is really open to that. So, my style of teaching suits this course if that makes sense (Teacher 8)

7.3.1.3.3 Classroom Management

It is important to note that the classroom management issue is complex, and teachers reported other variables as influencing students' behaviour (Erdogan *et al.*, 2010). The learning environment (see 7.3.2), students' mindset (see 7.3.2), teachers' confidence, group work and previous classes were also suggested as reasons by teachers. Of note, Teacher 7 reported that the presence of a Special Needs Assistant (SNA) in the class helped with one student's behaviour and that the SNA relayed that she felt this teacher was getting the effects from a boisterous previous lesson. Teacher 7 also reported difficulty with group work, specifically the organisation of groups. 'I found it hard to organise the students into groups so that all could take part.' (Teacher 7). Teacher 6 outlined how her previous teaching style in the computer lab was very individualistic: 'In computers, I wouldn't normally do group work. I'm like, you're at your computer, no talking, you're doing what you're doing'.

7.3.1.4 Teachers' Reactions

The last subtheme captured under 'Teachers' Experience' is 'Teachers' Reactions'. This sub-theme is concerned with evaluation, specifically the effectiveness of the course (Guskey, (Effectiveness)). Data from the workshop captured teachers' first reactions to the course in relation to the materials and content. Teacher 2's reactions were previously captured in the prototype of Version 1.

This section concerns reactions captured whilst teaching the course. Overall the reactions from teachers regarding the course and its teaching were very positive. Teachers used evocative language to describe their teaching. A snapshot of their quotes is captured in Figure 7-4.

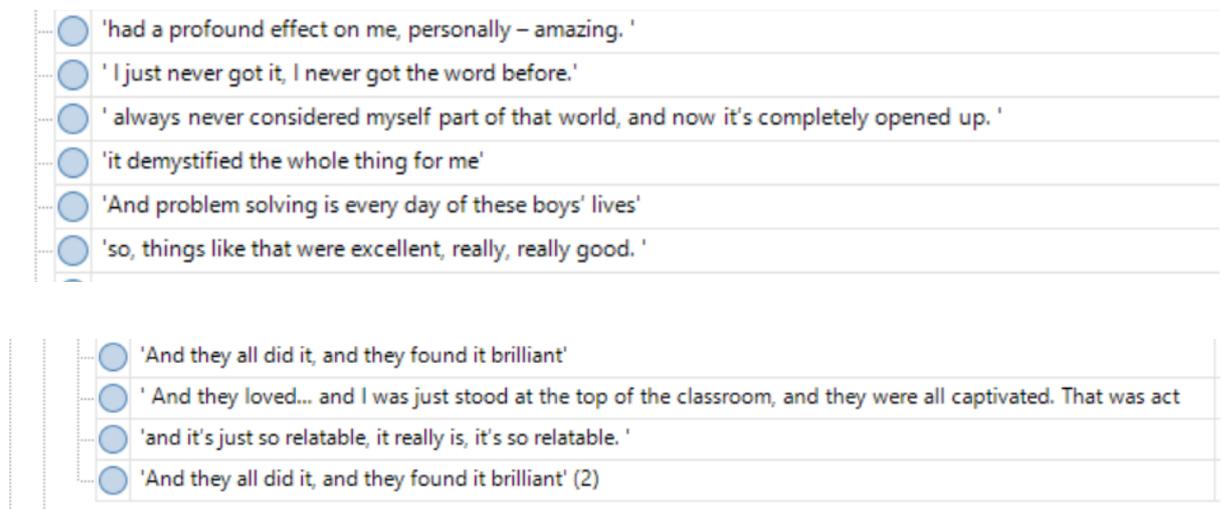


Figure 7-4 NVivo screenshot of teachers’ quotes in relation to the course

This sub-theme (‘Teachers’ Reactions’) overlaps with other themes, such as ‘Teachers’ Learning’, ‘Readiness and Sustainability. This section is concerned with two new sub-themes: ‘Teachers’ Confidence’ and ‘Students' view on computers’.

7.3.1.5 Teachers’ confidence

A growth in confidence was a strength described by teachers about this course. The confidence experienced was multi-faceted in that it stemmed from many sources and resulted in changes of behaviour.

Teacher 8 described his initial fear of Computer Science and Computational Thinking and how he then undertook a home technology project on completion of the course.

It’s just this big word and I think sometimes computers – even my interview for the space with computers, even all this terminology is like – it frightens people away from being users when it’s nothing it really opened my eyes to computers and to how user friendly they can be and I just wasn’t the most tech savvy person if I’m being honest Colette and just the confidence from even just teaching computers for the last three months, you know. I’ve got a projector now hooked up at home as a present for the kids for, we’re going to try and

create a cinema room. Something that I would have known nothing about prior.
(Teacher 8)

Teacher 7 highlighted her new confidence in this subject area. She had previously taught a Computational Thinking course but never understood exactly what the word meant. She has previously worked as an analyst where she wrote the requirements and steps for processes that were then coded. She had failed to relate these experiences with Computational Thinking. She now describes herself as a fellow Computational Thinking evangelist.

I have to say to you, the session you did with me has had a profound effect on me, personally – amazing. And I know it sounds a wishy-washy, but honest to god, in everything, how I think about everything, how I... done in my work, I just never got it, I never got the word before. (Teacher 7)

Yeah, but it's... again, you'll never... you can never assess that either, but it has... I was just closed off a bit to computers, because I'm a business, finance person. And I just suddenly... it demystified the whole thing for me. It's amazing, what that day did for me. As I said, I taught that course last year, I had no... we did it through the ECDL book, we had these notes, we did those points with the kids. And I never got that point, I never got how it's in everything we do, you know what I mean. So I know you're a bit evangelical, I'm a little bit like that now about it. . (Teacher 7)

Teacher 9 and Teacher 6 outlined how their confidence grew as they taught the course. Teacher 9 also reflected on how teaching something new can be daunting but was proud that she had the confidence to do that. She also stated her delight in how the shared experience of teaching the course with her colleagues brought them more together as a department. This was very apparent during school visits, where teachers shared their experiences and took on board successfully strategies from each other.

7.3.1.6 Students' views on computers

'they assume the computer can think and understand and make the right decision, you know.' (Teacher 7).

The second notable item captured under teachers' reactions was their surprise in relation to students' views on computers. Some students had a god-like opinion of computers. They believed that AI assistants such as SIRI and Google Assistant were real people and that the

computer was always right. Teacher 6 described a discussion she had with her class during her teaching of week 4 (AI and CT) part of the course

[Teacher 6] .. they actually truly believe that Siri is a real person. And I was like ‘Do you imagine her sitting inside in a little plastic box?’, ‘Well no...but she is real like’.

[Teacher 6] She thinks but...

[Teacher 9] She thinks. Is that what they said?

[Teacher 6] Yeah, like but how does she think, she is not human. But she does think because she says ‘Well, mmmm.... let me think about that’.

Teachers highlighted that as the course progressed, they found themselves having a specific goal to ensure that students understood that humans wrote computer programs. This goal was cemented from week 1 when it became apparent that students were both equally surprised and sceptical when shown code on how a computer checks if a word is spelt correctly. Teacher 7 highlighted how it is so alien to students that a computer is just following instructions. Teacher 6 reported on an amusing incident when students were playing the Codebreaker game against the computers. She mentioned to two students that they should whisper their answers in case the computer is listening, and after she told them she was joking, they questioned her on her ‘incorrect assumption’.

7.3.1.7 Theme Summary

The effectiveness of the course was evaluated in relation to teachers’ learning and reactions. Unplugged activities were shown to be engaging and thought-provoking for both teachers and students. The activities were found to be memorable, thus aiding linkage to other subjects and computer programming. This also contributed to the sustainability of the course. For school 5 they recommended that the course be split in two, with Units 1, 2 and 5 being completed first. This would then be followed by their own programming course. Unit 3 and 4, would be completed after this.

Teachers’ new founded understanding of Computational Thinking facilitated an ‘articulation grammar’ for the usage of CT’s concepts in other subjects. The course’s reliance on verbal reflection was found to contribute to classroom management issues, with

more written reflection suggested in its stead. Students' view on computers being king '*Deus ex Machina*' highlighted an important goal for this course, that students need to understand how computers work, considering they live in a digital world.

Finally, the course's instructional design was shown to influence and impact the teaching style of the teachers.

7.3.2 **Theme 2: Learning Environment:**

The second theme captures the Learning Environment, specifically the physical setting of the classroom. This theme also encompasses the school's Computer Science culture and how this impacted the students' mindset on the Computational Thinking course. Student characteristics and computer skills are also discussed.

7.3.2.1 *Computer Room*

In both schools, the teaching of Computational Thinking was scheduled during Computer Science classes. This was problematic for two reasons, 1) the classes were conducted in the computer room and 2) students' mindset regarding Computer Science. All teachers concurred that teaching the class in the computing room was problematic, as the physical computers were too distracting for students. Teacher 8 described the situation as follows:

it's like going to a toy shop and telling your kids you can't use the toys. All you can do is just sit there and watch me talk about the toys, but you're not allowed touch them or use them (Teacher 8)

Where possible, teachers moved the classes to ordinary classrooms, this worked much better, but had the knock-on effect of affecting the completion of MCQ assessments electronically (see 7.3.3.2.1).

7.3.2.2 *Student Mindset*

The majority of the teachers in School 2 discussed the student mindset in relation to students' views on Computer Science classes. Computer Science in School 5 is a non-exam

subject, taught from first year to Transition Year. The work is individualistic. Students cover the following topics over three years: typing, internet safety, file management, shortcut skills, computer hardware classes, Scratch programming and the aforementioned classes related to Computational Thinking. Teachers highlighted how students consider Computer Science a ‘relaxed classroom’, a ‘doss’. They come to the computer room twice a week where they work away on their own thing. Computational Thinking was scheduled during this period, and for some students (specifically from the older classes), they wanted to return to being able to use the computers. Teacher 7 expressed the view that she believes this was a factor in some of her classroom management issues. Teacher 2 had issues with students being assigned work to complete during their computer classes.

7.3.2.3 Timing

Teachers also mentioned the timing of the lessons, all teachers in School 5 found the forty-minute period too short. They stated that especially after scheduled breaks, with roll calls, it could be five to ten minutes before you start the class and then with explaining the activities etc, the forty-minute period is too tight. This situation resulted in teachers taking a minimum of two weeks per topic instead of one. Thus the five-week course became ten weeks for some teachers. Teacher 8 taught fourth years (TYs) in a double class, and he was able to complete the course in the allotted time. However some teachers highlighted how it is difficult to get double classes in the timetable as in their school this would result in Computer Science being in the same band as Science. Teachers also alluded to the fact that the pre and post tests before and after each topic ate in their short time, but this would not be a requirement for them going further (see 7.3.3.2.1) . Teacher 2 highlighted how with his Transition Year students, he was ‘happy to allow activities to proceed longer than anticipated’ as he did not have the time constraints of a ‘traditional class setting’. However, he stated how: ‘Attendance is definitely one of the bigger obstacles running this course

with TY's due to the high level of content/out of school activities that is organised for them'.

7.3.2.4 Theme Summary

The classroom environment was shown to be a relevant factor in the course's teaching and learning, with the computer room being at times distracting for the students. The course was shown to work best in a standard classroom, with teachers suggesting that they believed an 80 minute class would work better than a 40 minute class, although they recognised that they would not have to pre and post tests on each topic, which ate into time.

7.3.3 Theme 3: Students' Experiences

The third theme was 'Students' experiences'. Teachers' perceptions of student engagement, reactions, learning and assessment are encompassed in this theme. Similar to Version 1, teachers found that students were very engaged with the activities. Student learning and assessment are also discussed here, with teachers expressing their opinion on how best to assess this course. Elements of this theme overlap with themes such as 'Classroom Management' and 'Course Design', and thus where relevant, links are made to these sections.

7.3.3.1 Engagement

This sub-theme considerably overlaps with both the 'Course Design' theme and 'Teachers' Experiences' theme, and as a result, an overview of engagement is only presented in this section. (Detailed feedback on students' engagement with each activity is captured in section 7.3.4.) To summarise, all teachers agreed that students engaged positively with activities and content. This is best illustrated by using the descriptive statements teachers relayed, a sample includes 'a big hit', 'all students were engaged with it and enjoyed it',

‘interested’, ‘really engaged’, ‘intrigued’, ‘the boys LOVED the puzzles’ and ‘they were leaving the room talking about them’.

7.3.3.2 Student Learning

This sub-theme captured data that referred to student learning. Teachers highlighted the importance of ensuring student learning was balanced with their enjoyment of the activities. This was achieved by starting each lesson with a recap. Teacher 7 documented how with the card trick activity, her third years made the link with algorithms but with her first years she was cognisant that the lesson was more about enjoyment than theory. Thus a recap of theory at the next lesson was necessary. Teacher 10 concurs with this view. He alluded to the fact that they are an academic school and their students are used to memorising material. He highlighted the need for homework and student handouts as it provides a repetition of content. The majority of teachers stated that students were able to connect decomposition, pattern matching, and abstraction to ‘categorising buttons’. Teacher 10 was mindful that there may be students who do not make the link, and homework ensures this happens for all students.

7.3.3.2.1 Assessment and Learning

Students were assessed after the piloting of Version 1 of the course. The pre and post-tests provided findings on students comprehension of Computational Thinking, Artificial Intelligence and Algorithms. Students were also assessed on their views of the importance of logical and ethical thinking in relation to Computational Thinking. Students were also asked to state what they learned during the course (see 6.6.1.4.3). In this phase of the course, an assessment plan was devised for each unit (see Appendix L), influenced by Bloom’s Taxonomy (1956). Multiple choice questions formed a part of this assessment plan. It was envisaged that these tests would be issued online pre and post unit. It was also envisaged that these test would be ‘simple, short and quick’ as recommended by teachers in Version 1. The MCQ proved to be problematic: first years did not have email accounts.

Due to the ‘unplugged’ nature of the activities, some class forms were not scheduled for the computer room, and those that were scheduled were moved during the piloting of the course due to classroom management issues. The issuing of the tests also ate into the already limited class time: ‘The forms did not work when we moved the students to the computer room and we lost a good bit of time from the class. Without doing the post or pretest we started into the chili exercise’ (Teacher 10). The MCQ tests had to be printed out on paper. This resulted in an unnecessary administrative burden being placed on teachers: ‘Did post test on week 2 and pre test on week 3 - a lot of paper being used and not sure how seriously students taking it’ (Teacher 7). Teachers reported that some students specifically first years were uncomfortable with doing the pre-tests: ‘They completed the pre-test which they did on paper. I thought they were quite lost doing that which would hopefully change by the time they're doing the post test.’ (Teacher 9). ‘Students are also giving out - we’ve done this already why are we doing it again and I don’t know that they will take it seriously if they have to do it pre and post for each week?’ (Teacher 9). Teachers made the following suggestions in respect to assessment. Teacher 7 advocated for Kahoot

After the post test, students wanted to do a Kahoot quiz which they love. I found a Kahoot quiz based on computational thinking which worked really well on reinforcing the 4 main concepts - to my delight a lot of the students were getting the answers right so was a very easy and visual way to see that learning is happening - Colette - have you used Kahoot? My students love it and it might be something to think of using as a means of assessment?
Did post test on week 2 and pre test on week 3 - a lot of paper being used and not sure how seriously students taking it. Found it very difficult to get students attention

Teacher 9 recommended an assessment similar to the Junior Cycle Continuous Based Assessment (CBA).

Yeah, so that might work at the end, or getting them... You know, having them show something at the end of it all. Because they do CBAs now in all classroom subjects, classroom-based assessment (Teacher 9)

Teacher 2 in School 2 took on board the CBA idea and issued his students with a project (see Appendix M). He reported the average grade as being a distinction (80% upwards). The MCQ idea for student assessment was returned to for the Semi-Summative phase.



Pattern-Matching¶

•→ The patterns are similarities or characteristics that some of the problems share.¶

Example of Pattern-Matching¶

Computational Thinking

step of instructions on how to do something

Computational thinking I decided to pick the activity about attention to detail. Each step must be trying it out its done properly.

When making a sandwich and you said put butter spread or opened the container of butter would all this would be a huge mistake and therefore

Students were tasked in writing the instructions for a first. Every instruction is taken quite literally. First

Computational thinking is a set of problem-solving methods that are used to express a problem in a way which a computer could understand. It is necessary in computer programming to understand a problem as a series of small, absolute steps as a computer will not be able to carry out a vague command if there is any ambiguity in the instruction.

Computational thinking is based on four pillars or steps to breaking down the problem. These concepts are described as below:

Abstraction:

This is the process of removing unnecessary detail to only leave the key elements of the problem. This step is important to stop irrelevant details distracting the problem-solver from the structure of the problem.



Pattern Matching:

This step is looking for any repetition or recurring objects in a problem to further break the problem down. Often problems are just multiple smaller problems in a row. By solving a single instance of the problem, the solution can be extrapolated across the bigger problem.



Figure 7-5 A collage of Students' answers for a Computational Thinking project issued at School 2 by Teacher 2

7.3.4 Theme 4: Course Design

Theme 4 captures data related to the course design, specifically 1) the positive aspect of the course design, 2) recommendations for future versions and 3) changes that the teachers made to content specifically regarding making it more relatable for the students. Barriers to the course implementation, such as the computer room and lesson length, were also documented, and solutions provided to inform future versions. Specific feedback on the

individual unplugged activities was also gathered. Findings are tabulated chronologically into five tables, which contain the following headings (based on design concepts): pre-activation, reflection, theory, demo and application. Any issues encountered after each design concept are also tabulated in a separate 'Issues' heading. Similar to the tables in Chapter 6, (Table 6-3 to Table 6-7), the revisions of the design and content are validated extensively with teacher quotes.

The tables are displayed in landscape layout.

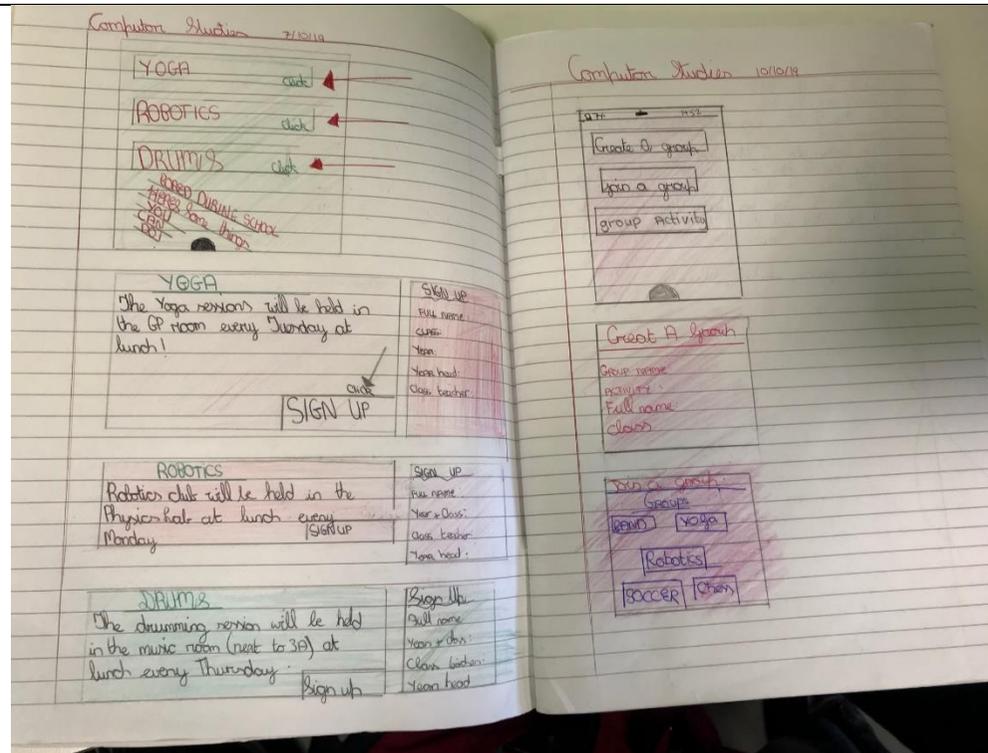
Table7-1 documents Unit 1. This is concerned with problem-solving, specifically generic problem solving and its subsequent relationship to the problem-solving framework of Computer Science, Computational Thinking.

.

Table 7-1 Summarises the feedback from Unit 1

Activities/Instructional Elements	Findings
<p>Pre-activation Puzzles Lesson 1 starts with two logic puzzles.</p>	<p>Teachers reported that the starter puzzles very much met their brief. They served as a warm-up engagement activity, highlighting that a strategy for problem-solving is better than guesswork. ‘Puzzles went so well some students who didn’t have consent form returned now really want to be part of it - the one student who continuously said their parents didn’t consent to it suddenly (!) found a signed form in their journal’ (Journal Teacher 7)</p>
<p>Issues</p>	<p>No issues reported with the Logic Puzzles</p>
<p>Pre-activation and Reflection Button Categorising and Thirty Second Game (see Appendix F)</p> <p>The purpose of these activities is to provide students with a practical application of decomposition, pattern matching and abstraction such that students and teachers can relate to and refer back to later on in the lesson.</p>	<p>Teachers used evocative languages to describe students’ engagement with these activities: ‘really enjoyed’ ‘really engaged’ ‘dying to have a go’.</p> <p>Two interesting findings reported. ‘Button matching was interesting because in one group one student was colour blind and couldn’t differentiate between green and red so I used this to have a discussion about the assumptions we make when we grouping - is colour the best category to use?’ (Teacher 7)</p> <p>The other interesting finding was in discussions with teachers at the end of the course, they outlined how they intend to use these activities as concrete reference points when they start teaching programming. This point was explored further in the ‘Teachers’ Reactions’ theme.</p>
<p>Issues</p>	<p>No issues reported with the thirty second game or button sorting</p>
<p>Reflection and Theory Explanation of the Computational Thinking concepts after the button categorising and thirty second game</p>	<p>The reflection part and the explanation of the theory that followed the pre-activation activities resulted in classroom management issues for over half the teachers. This issue was explored further in the ‘Teacher’s experience’ theme, as many explanations and solutions were put forward by the teachers.</p> <p style="padding-left: 40px;">We’re all seeing discipline issues where they are louder than usual and having to be reined in a bit every now and then. I’m overcoming this by getting them writing (always effective to calm a group down!) so every now and then they take some notes into their computer copies, usually to give me a break :) (Teacher 6, email)</p> <p>Whilst the placemat activity was used to get all students involved in the discussion. This did not seem to be as effective with younger classes as they were unfamiliar with it, and it took time to explain.</p> <p>A suggestion was made by two teachers to introduce writing activities as part of the reflection. and also as a homework activity.</p>

Issues	Placemats and Classroom management issues
<p>Demo video, exemplifications of decomposition, abstraction and pattern matching</p> <p>Real-life examples of decomposition, abstraction and pattern matching were introduced to students using video, e.g. facial recognition algorithms that use patterns, Ted Talk short video on the design of the London underground map (abstraction).</p>	<p>Video proved to be one of the positive findings from this study. Videos were used in many guises in this course from setting the hook for an activity, to being used as a teaching tool. In this lesson they were used to exemplify computational components and to both spark and foster discussion, hence contributing to the integration/reflection part of the design.</p> <p>‘The Ted Talk video is a great video, students were engaged and its very relevant’ (Teacher 2, email)</p> <p>A positive finding with reference to the examples used is that some teachers where appropriate, changed the ‘course examples’ to more localised example.</p> <p style="padding-left: 40px;">Underground video was good as was the discussion - I got the students on a page to map out how they would approach this task for a light rail system for <city>- it was interesting and useful to get them to think about what is the most important points and whose viewpoint is most important (Teacher 7, journal)</p>
Issues	<p>The Exemplifications were made local for example a local Light Rail app, Cinema App instead of Dublin Bus App.</p> <p>Homework was mentioned as part of this task. Teacher 10 got his first and second year students to design an app that would allow students to join three school clubs. He wanted to consolidate the decomposition, abstraction and pattern matching.</p>



It was pointed out that care must be taken to ensure homework does not eat into much time in the next class. Teacher 10 highlighted that projects would give the students focus. He also provided feedback on the students' work stating that: 'Their language could have been refined and many included some unnecessary noise like place your advertisements here but they got the idea of decomposition and abstraction.'

Application

Spelling Test

Students are asked to apply their knowledge of Computational Thinking to a scaffolded group activity. This scaffolded activity had two parts 1) students used CT to create an unplugged spellchecker and 2) watched a video that

The unplugged activity was reported as working really well but teachers were apprehensive with showing the video, as it involved code. This resulted in two teachers showing the output from the code, rather than running a video that showed the code

And then when I able to show your sample, this is actually what it's doing. And they just can't get their head around... they just never thought that that's what is happening .. It's keep reinforcing the same point, but it's so alien to them, that the computers are following instructions that are logical, that... because they definitely just think computers know everything. I suppose the way I say it to my own kids, ask Mr Google, as if Mr Google is a person that does things. (Teacher 7, group interview)

demonstrated how their solution is used by a computer.	
Issues	Explain more in the teachers' guideline document what the screencast shows on how the Java word checker works

In summary, findings from Unit 1 validated the dual purpose of the opening puzzles. 1) as warm-up activities and 2) as pre-activation of knowledge activities. The findings also validated the success and high engagement of button-categorising game and the thirty-second game. Teachers reported issues with the verbal reflections component of the design, and recommendations were proposed for written reflections. These proposals to the course design are discussed in detail (see 7.3.1.4). A similar proposal was suggested in Version 1 with the use of placements. This strategy was not as successful in this pilot, as students were not familiar with placemats. Teachers also suggested the need for homework, so as to consolidate the learning, support reflection, and provide feedback. Video as a teaching tool proved to be successful with both students and teachers alike. The need for localised examples was also shown to be necessary.

Table 7-2 is concerned with illustrating the finding from Unit 2. This unit was concerned with algorithms and also with logic.

Table 7-2 Summarises the feedback from Unit 2

Activities/Instructional Elements	Findings
<p>Pre-Activation Chill Chocolate game Card Trick</p>	<p>Once again the two pre-activation activities were successful in meeting their goals of 1) being an engaging warm-up 2) serving as pre-activation activities and 3) being instruments to learn the characteristic of algorithms before the theory is introduced. They were universally liked.</p> <p>The majority of teachers highlighted how the students were able to link the activity to algorithms. One teacher reported classroom management issues with her third years: ‘Some issues with messing with the sweets and the cards etc. At times difficult to police and manage across the room’ (Teacher 7, journal). Teacher 8 had a different experience with his second and third years, he used students’ pencil cases for the chilli chocolate game. ‘So, everyone had a pencil case, so between them all, they were able to come up with 13 pens and one rubber, and that was the chocolate and chilli. So, they all went back and did it themselves. And they all did it, and they found it brilliant. And I went around then and supervised, so just like a normal classroom’.</p> <p>Having all students involved, instead of the teacher and one student may be the reason for this.</p>
<p>Issues</p>	<p>Highlight the modification for teachers to use student pencil cases, for the chilli chocolate game. This ensures all students are involved.</p>
<p>Theory and Demo Video: Exact Challenge</p>	<p>The characteristics of an algorithm are now discussed. Videos provided the means to set the scene for an activity. The ‘Exact Instructions Challenge’ video (Darnit, 2017) was the lead-in to the Emoji Activity</p> <p>‘The peanut butter you tube clip was great. Gave the boys the right idea of the task I was giving them.’ (Teacher 9) ‘</p>
<p>Application and Reflection Emoji Activity</p>	<p>This emoji activity when conducted in the schools highlighted three issues, 1) younger students got disheartened with perceived failure in this task and 2) had difficulty articulating their instructions and 3) teachers were unsure what they as the ‘computer’ should know as they drew on the board.</p> <p>The task was found to be ‘too unstructured/too difficult for first years’ (Teacher 7) as they struggled to articulate their instructions. Students also became downhearted when their instructions proved to be inaccurate when the teachers drew them on the board.</p> <p>A similar issue arose in the DEIS school in Version 1, which was overcome by changing the format so that students were given the chance to change their instructions as they called them out to me as I drew the emoji on the board. This is probably the best way to teach this activity, as later groups are able to change their instructions based on what happens with the starting groups (Teacher 2).</p>

	<p>All teachers highlighted the learning involved with this activity and the importance in highlighting that computers just follow instructions.</p> <p>‘I actually think now, doing it with them, I’m thinking more and more, seeing their reaction to it, how important it is. Because as I said, they just... they are going to live in a world where it’s done, it’s handed to them with this fabulous interface with all their data and all of these things that they’re doing’ (Teacher 7).</p> <p>Teachers also struggled with knowing what they should interpret as they drew on the board, i.e ‘does a computer understand the word draw or circle for the basis of this activity?’ (Teacher 7).</p>
Issues	<p>The teacher guide book was changed to highlight that teachers should allow students to correct answers when giving instructions.</p> <p>Teachers were advised to act like a high-level programming language in that it would understand the concept of draw a circle but would need the size.</p>
Demo Pre-activation Logic is introduced using a pre-activation puzzle	<p>A clip from die hard video set the scene for the logic puzzle, and also helped to stimulate students’ imaginations. This was very successful. ‘So then when they were doing this, I’d say it was nearly that they felt like they had this bomb as well and they were, you know, so they could actually, it was like drama and education or whatever pretending this so they were, they enjoyed it’ (Teacher 8)</p>
Theory and Reflection (Integration and testing)	<p>One notable point was relayed by Teacher 10 when students were reflecting on what Logic was. They linked Logic to writing essays.</p> <p>We spoke about deductive and inductive logic and linked it to writing argumentative writing in English. We made the link between the type of logic we use in an English writing piece and the fact that we have to go back and check if our logic is reliable and 'check out' if our conclusions are valid. As an English teacher I found this very useful and believe most in the class would have gotten something from the class. (Teacher 10)</p>
Demo Codebreaker (Demo)	<p>The demo with the computer proved successful as it highlighted to students that the computer was following ‘deductive’ rules when trying to break the code. ‘Students were all engaged in looking at the screen and guessing what the computer would do next. Some students started to see patterns in what the computer was doing’ (Teacher 7)</p>
Application and Reflection Codebreaker (Application) and Reflection	<p>This activity was enjoyed by both teachers and students.</p> <p>The templates were changed to make them easier for Version 2, and this seems to have worked as no teacher reported that students had difficulty filling them out.</p>

Issues	<p>One teacher reported issues due to how she split up the groups</p> <p>‘Students were grouped in mixed ability and I noticed that ideally I would have split students so that the 4 students struggling were grouped together so that I could work through it with them with more scaffolding. As it was, some very good students didn’t get to experience the codebreaker correctly due to shortcomings by their partner which left them somewhat frustrated. Overall all students enjoyed it and figured it out correctly. (Teacher 7)</p>
--------	--

A summary of week 2 is as follows: once again the two pre-activation activities were validated as a successful design element. A change in one of the activities was suggested for the chilli chocolate game where students use materials at hand, pens etc, instead of chocolates. The success of videos was also validated where in this instance as a means of setting the scene for an activity and stimulating students’ imaginations. Changes to how feedback was given in the emoji drawing game was also suggested, as students became disheartened on receiving negative feedback. Teachers highlighted how they made linkages with this subject to other subjects (see 7.3.1.1) and the importance of students understanding that computers are not autonomous being, they are machines that follow instructions (see 7.3.1.4). One teacher reported difficulty with the organisation of groups for the Codebreaker game, which is discussed in detail in section (7.3.1.3).

Table 7-3 is concerned with illustrating the finding from Unit 3. This unit introduced students to binary and linear search. It also serves to allow students to apply Computational Thinking to a group activity titled ‘Searching to Speak’.

Table 7-3 Summarises the feedback from Unit 3

Activities/Instructional Elements	Findings
-----------------------------------	----------

<p>Pre-activation Logic puzzle (weighing scales) a logic puzzle, which also serves to review and introduce the concepts of decomposition (divide and conquer) and elimination which will be explored more when we do a binary search.</p>	<p>Once again the logic puzzles were enjoyed by all students ‘They loved the puzzle. They love it.’ (Teacher 9) Although Teacher 10 observed a notable difference between his first and second years stating how the first years were lost with how to approach the problem</p>
<p>Issues</p>	<p>The puzzle may be have to be scaffolded more for first years, or another replace it</p>
<p>Demo Video Trailer of movie ‘Diving Bell and the Butterfly’</p>	<p>The video was added as a hook to the activity. This was reported by all teachers as a success, it was found to not only set the scene for the activity but added clarity and realism</p> <p>‘The whole story and video behind it was brilliant. It was a great hook and a great learning activity for what they were doing and to give the background.’ (Teacher 8).</p>
<p>Application Searching to Speak</p>	<p>Teachers reported how students enjoyed and were engaged in this activity. Students were divided into pairs and tried to guess the middle name of their partner. ‘They are intrigued by this activity and enjoy acting out the situation’ (Teacher 10)</p> <p>Teacher 8 emphasised the ‘real-world’ significance of the problem. They are intrigued by this activity and enjoy acting out the situation (Teacher 8)</p>
<p>Theory, Demo Binary Search</p>	<p>Teacher 6 reported that the binary search theory was too complex for her first year class, and worked better with second year onwards.</p> <p>Teachers also outlined different ways binary theory was explained to students. Teacher 7 using the alphabet from the Searching to speak slides. Teacher 8 used an example of searching for a book in a library. Teacher 9 used the David Malan’s (2014) video explanation on binary theory using a phone book and then got the class to participate using a hiding game. ‘I got half the class to stand up and told the first boy to hand a pen to any boy standing and I would find who had the pen with a certain amount of questions. Once I began another boy asked if he could try. He did very well and got the concept of <u>splitting up the problem.</u>’</p>
<p>Issues</p>	<p>No issues reported</p>

A summary of week 3 is as follows. Two of the activities were reported as causing difficulties with first years: the weighting scale logic puzzle and binary search theory. Video was once again validated as an excellent tool, for adding realism to activities and setting the scene.

Adding extra content (regarding Jean Dominique Bauby) to the teachers’ manual after Version 1 proved successful, as background information was required by teachers for this topic. Teachers used a variety of examples in the guide to explain binary search.

Table 7-4 is concerned with Unit 4, which introduces students to artificial intelligence and ethics. The goal here is to discuss if the thinking used to create algorithms should be moral as well as valid (logical).

Table 7-4 Summarises the feedback from Unit 4

Instructional Elements	Findings
<p>Pre-activation Gender Bias Puzzle</p> <p>To highlight that ones’ decision making and reasoning may be affected by biases</p>	<p>The puzzle resulted in mixed results. Some teachers stated that the younger classes got the answer straight away, other teachers reported the opposite. ‘ Class went very well overall. Started with puzzle re mother - students got it very quickly - not sure this puzzle works too well with younger generation - linked it to ethics Activities but felt it didn’t really work.’ (Teacher 7 email)</p>
<p>Issues</p>	<p>May need to research a different puzzle. A good linkage is the ‘bias’ of the calculated grades algorithm for A Levels (Smith, 2020)</p>
<p>Application and Pre-activation Eliza ChatBot This activity uses CT in its problem-solving but also serves to show the difference between a scripted chatbot and an AI chatbot.</p>	<p>This exercise proved to be successful. Students recognised the patterns in the Eliza ChatBot’s answers. The activity also served as a starting point to more advanced chatbots -which are in essence just excellent pattern recognisers. Teacher 9 described the questions and answers the students used in the class when ‘talking’ to Eliza</p> <p>And I’m feeling happy. No, I’m feeling depressed was the first one and then I’m feeling happy and straight away one of them said to me, ‘She’s just repeating the pattern because she’s using the same words’, and I was like exactly. (Teacher 9)</p>

Notable Point	Teacher 8 did this class in the computer room and described how students were dying to test out Eliza themselves on the computer.
Application Chatbot	Teachers expressed initial apprehensiveness concerning this activity, but it went better than they expected. Although two teachers outlined that they would need to scaffold this exercise more in the future. An interesting discussion that resulted from this exercise surrounds the issue of slang and chatbots understanding of same. For this exercise, some teachers selected the topic for the chatbot rather than letting students decide themselves. Initially, Teacher 7 let the students decide on a topic , but for a different class she selected the topic of a cinema chatbot. This was due to a group conversation (with researcher present) when one teacher related how this strategy worked well for them.
Issues and Notable Point	Update students document to suggest an example such as the cinema app, and do an example of this instead of the sample Harry potter chatbot. Teacher 7 asks her students how many were interested in Harry Potter and only three raised their hands.
Demo Show code for scripted Chatbot (Demo)	There was an issue running this code (i.e. a syntax error), and thus it wasn't shown by teachers. This has now been rectified
Demo AI assistant and phone (Demo) Teacher Demo of AI in real-life. This shows the “power of computational thinking.” This was played after ELIZA to show contrast	This video was a resounding success, all teachers reported students engagement and inquisitiveness regarding this topic 'Students really liked the google assistant video - full attention and plenty of opinions as to their reaction to talking to robots in their job and the emotion response to this.' (Teacher 7)
Notable Issue	Some teachers changed the order of this activity, and showed to students after ELIZA for contrast. At the same time of this lesson, the robot Sophia was gaining media attention. She was interviewed on This Morning with Phil and Holly (This Morning, 2019). Teachers shared this resource with each other.
Demo Online 20 questions	Teachers reported the success of this 20Q activity, and how the students were particularly interested in the assumptions/contradictions that were made. An interesting point regarding the students view on the ubiquitous of the computer was observed: 'the boys played 20q.net, a few of them thought computer might 'hear' their item so they were whispering it to each other!! I was surprised that they're even thinking along those lines' (Teacher 6). With the widespread availability of AI assistants in most communication devices, this observation is not surprising, but is a little bit disconcerting.
Pre-activation	As before, the engagement and pre_activation puzzles proved to be successful and thought provoking

Trolley problem	
Demo Car example	Teacher 8 observed how when the trolley problem was re-imagined with students being the car drivers, and how for students and himself ‘ethics went out the window’. Thus, highlighting how our biases affect our decisions. All teachers reported the success of the driverless car video and students’ interest in the subject
Applications Moral examples: Moral machine.	Teachers reported that they did this exercise with students but not much notable feedback beyond that. Although Teacher 7 reported that she had interesting discussion with students in reference to saving the cat over a person etc., and Teacher 8 observed that students found them interesting and picked the options that saved most lives
Reflection (Integration) of topics on Power, AI, and health	Teachers reported great discussion on topics. Three notable points by teacher 7 were related to students understanding of AI and students themselves contributing content. Teacher 7 outlined how her students brought up and were very opinionated on the use of aimbots in Fortnite and how this resulted in a lifetime ban for the YouTuber FaZe Jarvis from Fortnite. She also spoke about their understanding: ‘Through questioning went back over artificial intelligence - students gave impressive answers that demonstrated their understanding of AI being only as intelligent as the computer programme’ (Teacher 7). Lastly she spoke about classroom management in relation to students determination on being heard. ‘A lot of student interaction in this class with students’ hands up through the class nearly to the point of not allowing me to progress.’
Issue	Only one teacher gave comment in relation to this video, and that was in relation to the verbal discussions resulted in classroom management issues.

A summary of week 4 is as follows. Students provided useful examples to use for content for this topic. Their suggestions included, aimbots used in Fortnite and the robot Sophia. Teachers successfully collaborated and shared these examples with each other. The scripted chatbot exercise was changed, to use an established topic rather than a student-created one, for younger classes. The gender bias examples was also shown to be ineffective with younger students, as they successfully stated the answer was the mother.

Table 7-5 is concerned with Unit 5. The purpose of this unit is to introduce students to the foundations of programming, specifically pseudocode. The goal of this lesson is to link Computational Thinking and programming.

Table 7-5 Summarises the feedback from Unit 5

Activities/Instructional Elements	Findings
<p>Pre-activation One Logic Puzzles To engage students and to introduce coding/debugging</p>	<p>Teachers skipped this puzzles. They were behind on their schedule and were determined to finish the course before Christmas, thus went straight into the five fundamentals and cup of tea example.</p>
<p>DEMO Video of five concepts of programming</p>	<p>‘They really enjoyed the Big Bang clip and this was very useful when describing infinite loops and input/outputs. The 5th week both personally and for the students was overall a little heavy and not as interactive as the previous weeks. But all in all it flowed pretty well over the 2.5 classes I spent on week 5’. (Teacher 8)</p>
<p>Theory and Exemplification This activity uses a cup of tea to explain the five concepts</p>	<p>Teacher 7 changed the cup of tea idea to a barista in Costa. She reported how this analogy worked well.</p> <p>I related it to someone serving coffee in Costa or a coffee machine and how each person asks for something different - being the variable which is only relevant to the server until they get their coffee and is then forgotten - was also useful in the decision of tea/coffee - the computer/server doesn’t know in advance what the answer will be - was also a good analogy in terms of sequence - do you ask if they want sugar before you ask what they want and why not - students seemed to get the concept and it made it a lot easier to explain the algorithm. Students in both this class and last class asked does the computer remember what you wanted the previous time and use this information - good link to AI and how computers over time may get more intelligent. (Teacher 7)</p> <p>Teacher 7 also reported that she used the envelope example to explain variables which worked really well. ‘I used the envelope to explain variables Yes/No for sugar and I thought it worked well as a first explanation for variable - students were not making the link to maths variables automatically but giving them this link helped them I think’ (Teacher 7).</p>
<p>Application Write Pseudocode 1: Did I Pass?</p>	<p>Students were unable to write pseudocode for this example. Teachers outlined how students were unable to come up with answers themselves, even when given sentences to use.</p> <p>Teacher 7 outlined the following interesting points regarding the ‘Did I pass activity’, which got me thinking about the PRIMM model. She stated the following 1) students needed more scaffolding, 2) they were overwhelmed, 3) they lacked a frame of reference for finished result 4) she needed to go at a much slower pace and 4) she ended up going through the finished result on the board with the students line by line.</p>

	<p>This weekly feedback coupled with feedback from Teacher 6 and Teacher 8, resulted in me changing this topic before Teacher 9 and teacher 10 started it.</p>
<p>Issue</p>	<p>The problems reported by teachers resulted in me sitting down with Teacher 9 during a visit and devising a way to get students to interact with code. Their interaction was influenced by the PRIMM model. Feedback from Version 1 from students, indicated that they wanted to use code rather than pseudocode. The PRIMM model formed the instructions of my activity (i.e. the application part)</p> <p>Instead of getting students to use cheat sheets to write the pseudocode for a programme that calculates if a student passes an exam. Instead they were given the code, after the five fundamentals had been demoed. They were asked to type the code into an online python compiler. Next they were asked to predict what they thought would happen, to identify the relevant fundamentals of the code. To run the code, and then change the code to have a different pass value.</p> <p>Then we decided to introduce Python using an online compiler. Even though the teacher was not familiar with Python, she was able to understand the code of Did I Pass? And felt confident to try it out.</p> <p>Feedback from Teacher 9 who tried this out</p> <p style="padding-left: 40px;">We started with a recap of the 4 elements in computational thinking and then went on to cover the slides. We spoke about input, output etc. This worked well because when we opened up our python compiler for looking at the 'Did I pass' activity the students were able to pick out the lines of input and output. All students were on task . (Teacher 9)</p> <p>Further feedback highlighted the following. The need for first-years to have access to a soft copy as older classes code as a lot of syntax errors</p> <p style="padding-left: 40px;">I got them to type the code, instead of a copy and paste. For the 'newbies' it helped to bring home the point that you have to be accurate with your syntax and especially with python with your indents. Before they wrote it out, I asked them to predict what they thought was going to happen. Then type the code and run it. I was very surprised how many had syntax errors and getting friends to help each other and spot the mistakes helped greatly. (Teacher 9)</p> <p style="padding-left: 40px;">'We did all three problems. After each problem I asked them to link the code to the five fundamentals (where relevant)' (Teacher 9)</p> <p>As researcher, I subsequently developed two other activities that fitted into the PRIMM and five fundamentals</p>

	<p>model. Second activity, resulted in students changing the pass grade and the third activity to copying the code 5 times to highlight the advantages of Loops. Teacher 9 piloted these activities with her students and highlighted how they were a great segue to loops, and how she was able to link this exercise back to the Big Bang video</p> <p>Two notable comments she emailed me were ‘I was very impressed at the end.’ ‘They really enjoyed inputting the code and seeing it work.’ (Teacher 9)</p>
<p>Application Activity: What is the average grade</p>	<p>This wasn’t done using pseudocode as students found the jump from Cup of tea to Did I pass too much? New examples were written for python.</p>
<p>Teacher demos using weighing scales that computers think in twos</p>	<p>This wasn’t done using pseudocode as students found the jump from Cup of tea to Did I pass too much? New examples were written for python.</p>
<p>Application Activity: What is the max number? Write Pseudocode</p>	<p>This wasn’t done using pseudocode as students found the jump from Cup of tea to Did I pass too much? New examples were written for python.</p>
<p>Demo videos</p>	<p>The videos on pseudo code and Caesar Cipher were really good and the students understood a lot more from these. They really enjoyed the Big Bang clip and this was very useful when describing infinite loops and input/outputs.</p>
<p>Notable Comments</p>	<p>‘The 5th week both personally and for the students was overall a little heavy and not as interactive as the previous weeks. But all in all it flowed pretty well over the 2.5 classes i spent on week 5.’ (Teacher 8)</p>

A summary of week 5 is as follows. Once again, videos proved to be a successful teaching and learning resource, from the Big Bang clip on algorithms, to David Malan's summary of pseudocode and the five fundamentals. Although the cognitive load of pseudocode initially proved too much for students, the PRIMM model being incorporated into Python examples allowed the teaching of coding in a 'Low Threshold High Ceiling' way. This change in the teaching of topic 5 was also introduced into the teaching of the semi-summative phase, where I ensured I had a soft copy of the code available for all students in case of repeated compiler errors due to inaccurate copying. Some students will have little to no programming experience, and compiler errors will significantly affect their confidence (Connolly, Murphy and Moore, 2009).

7.4 Chapter Summary

This chapter set out to document the teacher workshop's findings and the prototyping of Version 2 of the course. The workshop proved to be a successful means of imparting information about the course to the teachers. Data generated from the course confirmed that the course's content met their needs and that they felt confident in teaching the course. They also confirmed the appropriateness of the course for Irish second-level students.

Four themes were generated from teachers' qualitative data captured during and after their teaching of the course. The themes were 'Course Design', 'Teachers' Experiences', 'Learning Environment' and 'Students' Experiences'. The themes provided answers to questions on the practicality, low threshold, quality, and effectiveness of the course. The course content, such as the teacher guide and course materials, was shown to be both usable and appropriate for teachers. Teachers also provided validations and recommendations to the local instructional design of the course. The warm-up puzzles and activities were shown to be engaging and thought-provoking for both teachers and students. The activities were found to be memorable, thus aiding linkage to other subjects

and computer programming. Teachers' new founded understanding of Computational Thinking facilitated an 'articulation grammar' for the usage of CT's concepts in other subjects. Video proved to be a multi-dimensional tool, as it aided demonstration, imagination, teaching and provided a hook to many of the unplugged activities. The course's reliance on verbal reflection was found to contribute to classroom management issues, with more written reflection suggested in its stead. Homework was also suggested as a tool to aid reflection. The classroom environment was shown to be a relevant factor in the course's teaching and learning, with the computer room being at times distracting for the students. Finally, the instructional design of the course was shown to influence and impact the teaching style of the teachers

The course was shown to work best in a standard classroom, with teachers suggesting that they believed an 80-minute class would work better than a 40-minute class.

Chapter 8 is concerned with the semi-summative evaluation and trial of the Computational Thinking course (Version 3) and corresponding instructional design.

8 Semi-Summative Phase Version 3

8.1 Introduction

Chapter 8 concerns the semi-summative evaluation phase, specifically the final ‘trial’ of the Computational Thinking course (Version 3) and corresponding instructional design. The principal contribution of this research is Version 3 of the Computational Thinking course, and the ADAPTER framework, which can be used by teachers and students when learning, teaching and designing Computational Thinking courses.

This chapter is concerned with first outlining the evaluations of the semi-summative phase. These evaluations are then collectively discussed with the evaluations from Version 1 and Version 2. Their purpose is to answer the research question: what are the characteristics of a practical, engaging, effective, high quality, and low threshold course for both the learning and teaching of Computational Thinking to Irish post-primary teachers and students? This study also aims to validate whether unplugged activities can be successfully used to teach Computational Thinking.

8.2 Semi-Summative Phase

The semi-summative phase was conducted from November 6th to 17th Dec 2019, in School 1 and School 6. It overlapped with Version 2 of the prototype, which took ten weeks rather than five to complete. School 6 was new to the study. The participating teacher (Teacher 11) taught science, physics, IT and maths at an all-girl school. He had previously contacted me after the 2017 CESI conference for information on my study. I purposely contacted him in 2019, as I wanted more girls schools involved in the study. School 1 had previously been involved in the prototyping of Version 1 of the course. All students self-selected into the study: fifteen from School 6 and five from School 1. The students in School 6 participated in the course outside of school hours.

Table 8-1 School profiles for Version 3

School	Dates 2019	Notes
School 6. All-girl, 15 students, 5 th Year (10) and TY students (5)	Nov 6, Nov13, Nov 20, Nov 27,Dec 4	Teacher absent week 3. Interviewed teacher on 5 th December
School 1. All-boy, 5 TY students,	Dec 2, Dec 9, Dec 10, Dec 17	Different teachers observed classes

8.3 Student Profile

Fifteen students participated from School 6, with ten from 5th Year and five from TY.

Eight students had previous programming experience (Python and Scratch). Five students participated from School 1. Two had previous programming experience (Java and Python).

8.4 Content Experts

Two content experts reviewed the final version of the Computational Thinking Teacher’s guide. Both experts permitted their names to be used, but comments attributed to them are cited without a name. Expert 3 is an assistant professor at Maynooth University. He has been involved in Computational Thinking research for the last few years and is involved in a Computational Thinking outreach programme with Irish post-primary schools. He recently supervised a Masters student who also created a Computational Thinking course for post-primary students. Expert 4 is a teaching fellow at Queen Mary, University of London. She previously worked on the ‘Barefoot Project’ and was involved in writing the concept documents for same, and also many resources for primary teachers to implement Computational Thinking in one-off activities in their class.

8.5 Findings

Both qualitative and quantitative data was captured in this phase. This consisted of student pre and post tests, and interviews with one teacher (School 6) and two content experts. The findings are presented under the following headings: engagement, low threshold, effectiveness, and quality.

8.5.1 Engagement

Engagement was important for two reasons: 1) evaluation of course content 2) facilitation of learning and interest in Computational Thinking. Engagement data was captured during the pilot of Version 1 and Version 3. In the semi-summative phase (Version 3), student engagement data was captured after each lesson (using the engagement questionnaire). In addition, an interview captured the teacher’s perspective on engagement.

Results from Version 3 strongly indicate that most students found each individual unit engaging, specifically its contents and activities. The majority of students ‘Agree’ or ‘Strongly Agree’ that the course was not boring, they did not daydream, they were focused on the tasks they were learning, and felt the time went quickly.

In School 1, the median value for each of the eight engagement items was 4 or 5, indicating that most students ‘Agree or Strongly Agree’ with the eight engagement statements.

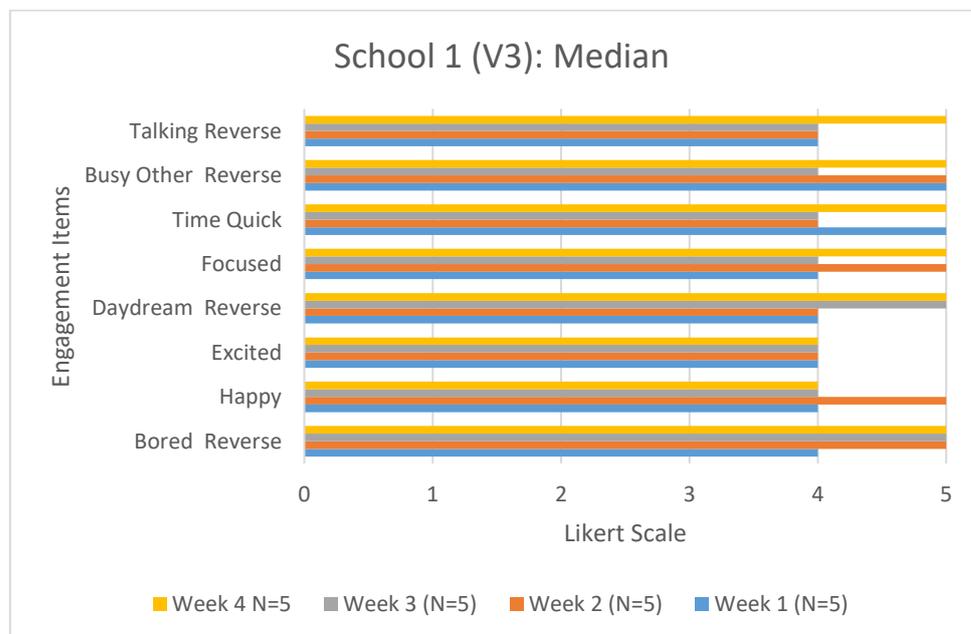


Figure 8-1 Median Engagement scores for School 1 (V3)

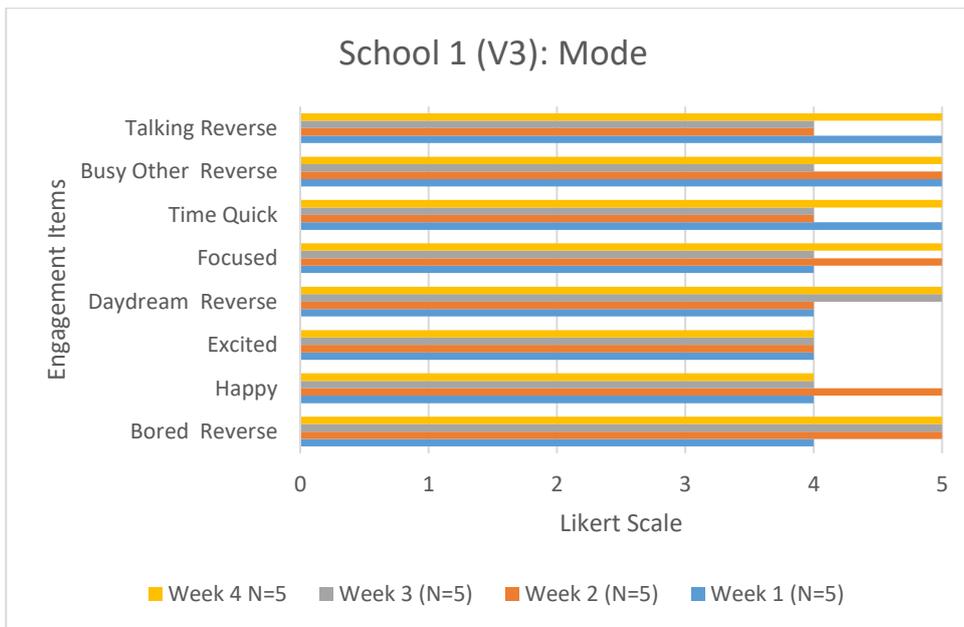


Figure 8-2 Mode Engagement scores for School 1 (V3)

Note: Week 3 Busy Others and Talking are multiple modes, smallest value shown.

In school 6, the median item for seven of the items every week was also 4 or 5. The last item: ‘During this lesson: I talked to others about stuff not related to what we were learning,’ had a median value that ranged from 2 to 4 over the course (one 2, two 3s, two 4s). This item was in connection with behaviour, where students indicated that they did talk to other students about topics not related to the course, during the lesson. An explanation for this may be that the course was conducted after school. My field notes confirmed that I did hear one of two students ask questions about homework during the activities.

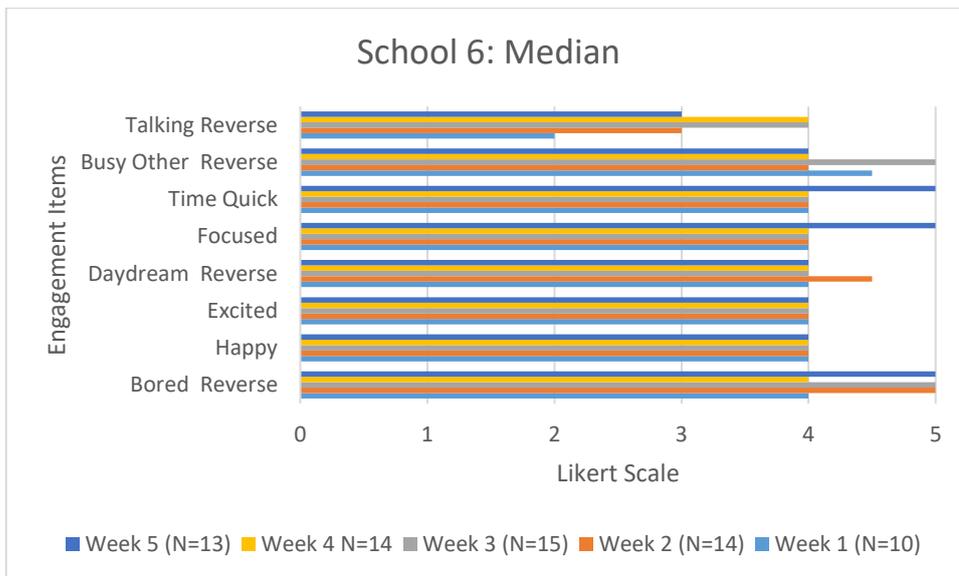


Figure 8-3 Median Engagement scores for School 6

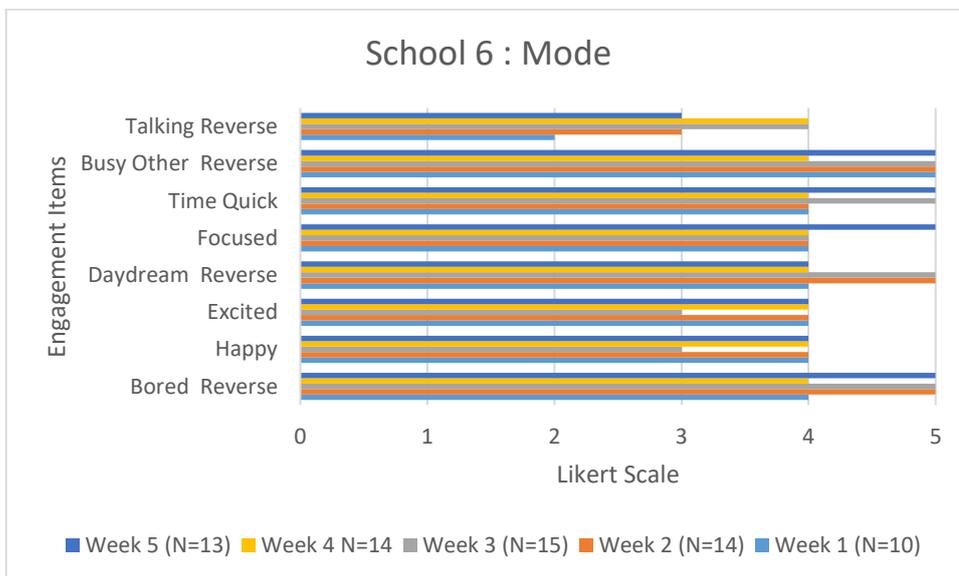


Figure 8-4 Mode Engagement scores for School 6

Note: Week 3 Happy and Excited, Week 5 Talking are multiple modes, smallest value shown

The interview with Teacher 11 also supports the finding that students in School 6 found the course engaging. The teacher brought up the topic independently before the interview had even started, and he was thus subsequently asked to repeat his views on engagement in the interview. He highlighted how the course was conducted outside class time, and students voluntarily stayed for an extra 80 minutes every week, with no loss of students. He also

stressed how impressed he was that on one week, his TY students stayed an additional three hours in school to attend the course, as their day had finished at midday. Teacher 11 attributed student engagement to a mixture of factors: course content, activities, group work and teaching methods.

Engagement	0	0
Stayed in school	1	2
'hand-on as well as minds on'	1	1
activities	1	1
set correctly	1	1
on task from day 1	1	1

Figure 8-5 NVivo screenshot of codes captured from Teacher 11's interview

I think it was very engaging. The activities were all engaging. They were in groups. They were hands-on as well as minds-on. You brought in plenty of props. It wasn't just things that was being done on a whiteboard or even on a computer screen. Students could actually, with the buttons, they could actually get tactile with them and that's very important in terms of engagement. (Teacher 11)

The activities, he believed, were very suitable in that they 'challenged the students', 'they weren't trivial exercises' 'they were also fun'. He liked the mix of real-world problems with puzzles that needed to be solved. He reiterated the importance of engagement as a characteristic of a course, as students have many demands on their time.

I thought they themselves kind of grew throughout the five weeks with the activities and that's basically why they kept coming back, because the demands on our students are vast. And unless they're engaged and it feels worthwhile, they're, you can say goodbye to them, you know, so that was proof, proof of the concept, you know. (Teacher 11)

The findings from Version 3 are consistent with the findings from Version 1, where results supported the view that both students and teachers found the course content interesting and

engaging. Teachers in Version 1 identified the following characteristics: competitiveness, clarity, lead-in hooks, interesting content and accessibility as impacting engagement. Teacher 11 also attributed student engagement to the activities and interesting content but also group work and teaching methods.

Students' positive engagement with the course and activities overall strongly suggested that the course content is age-appropriate and relevant. Students in the course highlighted how both the engaging content and activities helped with their learning and interest in the topic. 'I really enjoyed the algorithms and the Ethics section of the course. I'm usually very easily distracted and if I'm not interested in something, I don't participate well but I didn't find that happening, I really enjoyed this course' (Student S4). 'I really enjoyed the practical games and explanations that we got to do before the theory as I felt it helped to explain the theory and information we were given. I found it made the course more enjoyable' (Student A3).

8.5.2 Low threshold

Is the course low threshold? The answer to this question is multi-faceted as it concerns three key factors: resources, student prior knowledge of Computer Science and programming, and teacher prior knowledge of Computer Science and programming.

With reference to resources, the unplugged materials are inexpensive, they can collectively be purchased for under €35: playing cards, buttons, 30 Second Game, chilli and chocolates. The course was designed to use only technology found in a 'normal' classroom: teacher whiteboard, projector, and a computer with internet access. However, programming was introduced for Unit 5, as an alternative for pseudocode. This provided a dialectical dilemma as it affected the course's low-threshold component, specifically with reference to resources and teachers prior knowledge. Regarding resources, it may be argued that the course resources are still of low threshold for the Irish context as 1) computers are only

required for one unit, 2) the Python compiler is accessed using a browser and 3) in the current COVID climate, many schools in Ireland have updated their laptop supplies (Kelly, 2020).

With reference to students' prior knowledge, after Version 1, some students stated that they found Unit 5, the pseudocode component, difficult. This was not the case after Version 3. Coding was only mentioned twice in feedback, with both students saying they wanted more. Of the twenty students who completed the semi-summative phase, nine had no prior programming experience. The collective findings from Version 1 and Version 3 suggest that the course is low-threshold regarding students prior knowledge, especially when the results are triangulated with engagement scores. However, caution must be applied to these findings as the sample size (N=20) from the semi-summative phase is small.

Regarding teachers' prior knowledge, all the teachers involved in Version 1 expressed the view that they would be confident teaching the course, all the teachers involved with Version 2, stated they would teach the course again. There were two teachers with no Computer Science experience in Version 1, and two teachers with no programming experience in Version 2, with one of these teachers having no Computer Science experience. In Version 3, Teacher 11 had both programming and IT experience. He expressed the following view regarding inexperienced teachers:

But I think the way it was presented to the students, that the teacher, if they were using these resources, wouldn't have to have that level of familiarity with them because they were, they were practical, they were hands on and they weren't, they weren't jargon heavy. I think you introduced jargon where, when it was kind of necessary. And that was refreshing because, you know, if anyone picks up a computer programming book or, you know, a standard resource, you know, it's very, very jargon heavy, you know, and everything has to be defined, you know. And that's not just at the top of all of the programme, everything, you know, has a certain name and it's definite and it's discrete and there's no ambiguity in it at all.

But I think the examples and the tasks that you did and the way you approached it, it wasn't really needed, you know, and it would have taken from the task to be too

jargon heavy. Like the problem and solving the problem was at the core and at the front and I think that's what the students wanted as well. (Teacher 11)

Collectively the above findings suggest that the course can be considered of low threshold. However, with reference to Unit 5, Teacher 9, who taught this unit twice using Python had previous programming experience, and I taught Unit 5 twice for the semi-summative phase. As the students found no difficulty with the content, it seems likely that the teachers also would have no difficulty. However, this was not trialled in this study.

8.5.3 Effectiveness

The following section is concerned with evaluating the effectiveness of the course.

It includes findings concerned with student and teacher reactions, student and teacher learning, and student outcomes from the semi-summative phase. Findings from this phase are first discussed before they are triangulated with findings from the prototype phase.

8.5.3.1 Student Reactions

Students' reactions to the course were collected using the 'End of Course' questionnaire, which consisted of both Likert statements and open-ended questions. The open-ended questions were analysed using content analysis. The findings from the analysis are presented per school but discussed collectively.

8.5.3.1.1 Quantitative Data Student Reactions

The 'End of Course' questionnaire contained five items related to effectiveness. Using a five-point Likert scale, students were asked to rate their agreement from 'strongly disagree' (1) to 'strongly agree' (5) with the following statements:

- Overall, I would rate the quality of this course as outstanding. (best_evaluation1)
- Overall, I would recommend this course to other students. (best_evaluation2)
- This course was a waste of time. (guskey_satisfaction_2) (reversed)
- I am dissatisfied with the activities used in this course. (guskey_satisfaction_3) (reversed)

- I am very satisfied with the content of this course. (guskey_satisfaction_1)

The mode and median values for each of the five items per school are tabulated in Table 6-14 and 8-7.

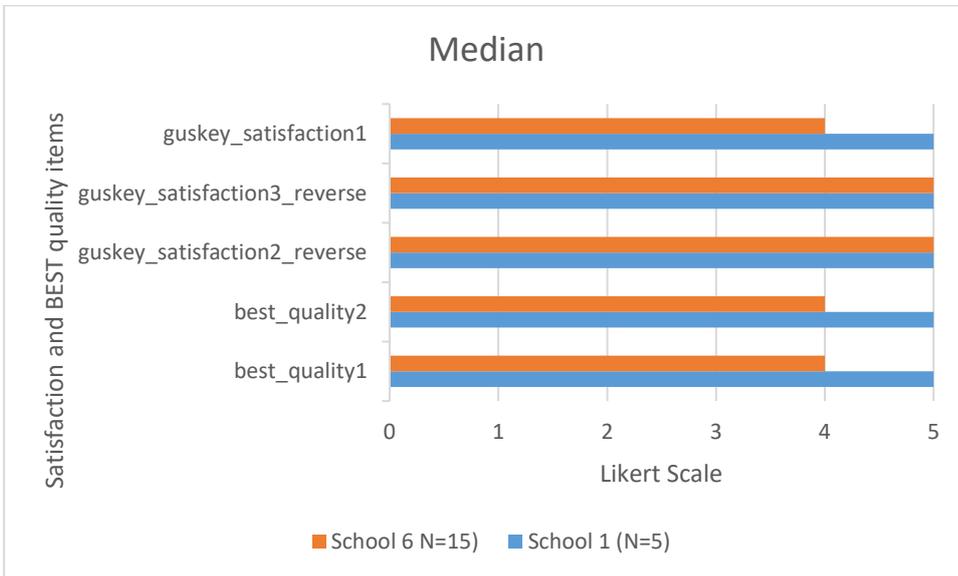


Figure 8-6 Median values from School 1 (V3) and School 6 showing Satisfaction and BEST Quality scores.

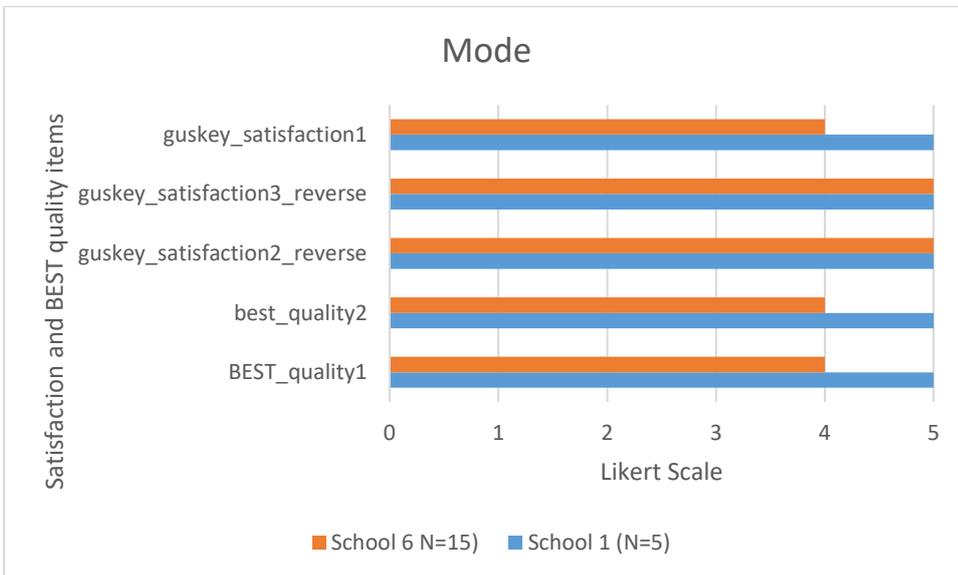


Figure 8-7 Mode values from School 1 (V3) and School 6 showing Satisfaction and BEST Quality scores.

The mode and the median value for each item related to the satisfaction is either 4 or 5 (see Figure 8-6, 8-7). This corresponds to Agree or Strongly Agree. It is apparent that most

students had a positive reaction to the Version 3 of the course. In short, they found the quality of the course to be outstanding, they would recommend it to others. They did not find it a waste of time and were satisfied with both the content and activities used in the course. In fact, the mode and median values for the above items were either 4 or 5 for all schools that participated in the pilot of Version 1 and Version 3 of the course

In the semi-summative phase, the above quantitative findings were also explored further by analysing the three open-ended qualitative questions. The answers to these questions were analysed using content analysis. The questions were as follows: Q1 What are the strengths of this course? Q2 Do you have any specific recommendations for improving this course? Q3 What do you think you learned during the course?

8.5.3.1.2 Strength

Table 6-15, 8-3 display the results from the content analysis of the question: ‘What are the strengths of this course?’ with examples from each content category. It also displays how many times the category was mentioned and by how many students.

Table 8-2 Results from the analysis of the open-ended question: What are the strengths of this course? (N=15, School 6)

Category	Description	Example	Frequency Column	
			Mentions	By Person
Media	Answers related to media, specifically the videos used in the course	<p>‘The videos and activities used to create a better understanding of the concepts we were learning.’</p> <p>‘the short videos to help explain the term used in Computer Science.’</p> <p>‘I loved the videos as they gave you a visual explanation and broke up the talking and activities well. It also was nice that it was catered to relate to us as it was videos implying the days learning objective through videos that we are familiar with,</p>	8	7

		eg. todays one was the big bang theory.'		
Group Work	Answers relate to group work	'the group activities were particularly engaging.' 'I loved taking part in the group activity because thats where i find my strength is'	4	4
Content	Answers related to the Course Content	'Its content is effective in most subjects, especially STEM subjects' 'It was very interesting and engaging but at the same time educational.'	4	4
Activities	Answers related to the unplugged activities	'I liked the hands-on activities, everything was explained clearly and I could understand everything' 'It uses a lot of fun activities and media to teach'	6	6

Table 8-3 Results from the analysis of the open-ended question: What are the strengths of this course? (N=5, School 1)

Content	Description	Example	Frequency Column	
			Mentions	By Person
Pedagogy	Answers related to the way the course was taught using activities, interactive, collaborative and the demonstrations	'the complex yet enjoyable and fun course with fun ways of learning and a great teacher.' 'it is really fun and keeps you active with tasks and questionnaires'	3	3
Content	Answers related to the Course Content	'Better understanding of how a computer works.' 'You can easily learn about logical thinking and being controlled by being given sweets.'	2	2

Media (specifically the use of videos), the course content, pedagogy, unplugged activities, and group work, were all identified as the course's strengths by students. These answers

resemble those expressed by students in Version 1. However, there is one difference, the number of answers that referred to the use of videos, 35% overall (50% in school 6). This is a promising find, as the number of videos used in the course increased from nine to fourteen, from Version 1 to Version 3. Notable additions include a clip from Marcus du Sautoy on facial recognition algorithms (Briggs, 2017), a Ted talk animation on the ‘Ethical dilemma of self-driving cars’ (Lin, 2015); a clip from the Big Bang theory on Sheldon’s friendship algorithm (Cendrowski, 2009) and a summary video on pseudocode from (TED-ED and Malan, 2013). Teacher 11 concurred with the influence of media, stating the following:

I think they enjoyed the videos as well because there was a kind of humorous side to it, not just Big Bang Theory or some of those but there was kind of a, you know, a kind of light touch. I don’t know did you select them for that but definitely there was, there was a humour to them. And even the Marcus_du_Sautoy video, I hadn’t seen that so, you know, and I have seen a lot of his work but, yeah, that’s all very good yeah.

The above data was also triangulated with a Likert item on the ‘End of course questionnaire’: Media used in this course (for e.g. videos, websites, slides) were helpful in learning. 93% of students in School 6 agree or strongly agree with this statement, and 100% of students in School 1(V3) strongly agree.

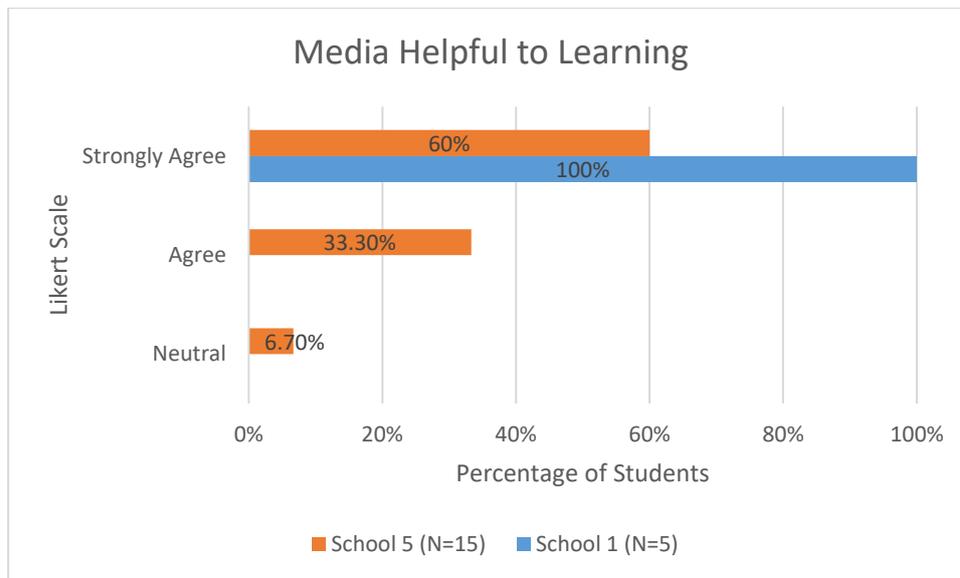


Figure 8-8 Bar Chart of students’ responses to ‘Media used in the course was helpful to learning’

8.5.3.1.3 Recommendations

In Version 1, findings from the question: Do you have any specific recommendations for improving the course? were used to inform changes for Version 2. In this phase, they are used to contribute to findings for students’ reactions to the course. The answers returned were diverse and very promising. Two students answered this question from School 1. Their answers were as follows: ‘No I do not’ and ‘maybe the course could contain more actual coding I felt the most enjoyment partaking in it’. The answers from School 6 are tabulated in Table 8-4. The two highest mentions were no changes and more activities. These findings suggest that most students did not have any major concerns or difficulties with the course.

Table 8-4 Results from the analysis of the open-ended question: Do you have any specific recommendations for improving the course? (N=15, School 6)

Content	Description	Example	Frequency Column	
			Mentions	By Person
More group work	Answers referred to group work	'More group work and letting us say our ideas out loud	1	1
Content	Answers referred to course content	'I would have liked to do more on ethics if possible' 'more coding'	2	2
More activities	Answers referred to unplugged activities	'a couple more activities' 'Add more activities like the chocolate bar and chili' 'More ability to participate more activities etc.'	3	3
Positive No Change	Answers specified that students happy with course the way it is	no no :) No I found it very fun Thank you	3	3

The above recommendations were triangulated with an item on the 'End of Course' questionnaire: 'In this course, I was able to get feedback on my work'. Only 46% of students agreed with this statement in School 6 but 100% of the students agreed with this statement in School 1(V3). Thus for future versions, homework will be offered to facilitate feedback.

8.5.3.1.4 Conclusion

Similar to findings from Version 1, the findings from Version 3 highlight that students had overwhelming positive reactions to the course.

8.5.3.2 Teacher Reactions

Teacher's 11 interview was coded using thematic analysis. The themes were coded deductively (*a priori*): Content, Reactions, Engagement and Low threshold.

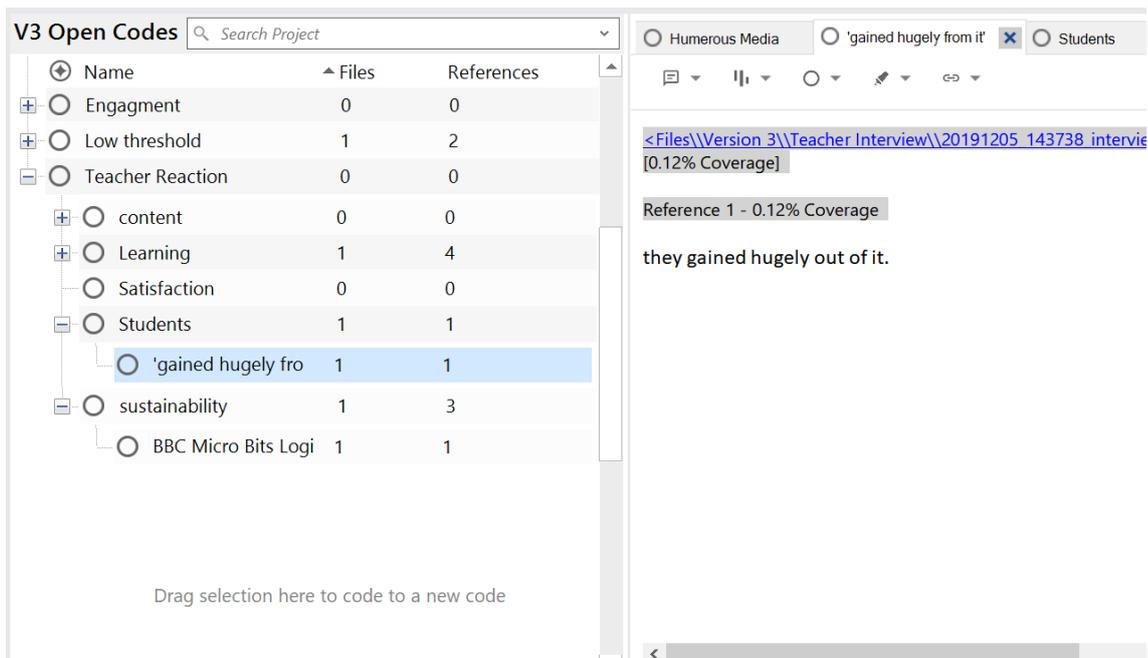


Figure 8-9 NVivo Screenshot showing the codes from the Teacher Reaction Theme for Version 3

Teacher 11, similar to the other ten teachers who were involved with this study, had a very positive reaction to the course. With reference to the course content, he stated that it was highly appropriate and that: ‘It came within their understanding and concept of what computers are used for, for say the problem solving.’ Regarding the activities, he mentioned the following:

I thought they were suitable in that they challenged the students, that they weren’t trivial exercises. I thought that there was a nice mix between say problems that they could identify as being real world and that needed to be solved. So, say the sorting problems as well as fun, I mean say the card tricks, you know, that was a really good fun activity but yet there was some really good algorithms behind it and like the code breaking games. (Teacher 11)

He also alluded to the fact that he (like all the teachers in this study) found the content very interesting and unique:

So yeah, I felt engaged and I learned the card trick and I was happy to engage as a student and I think that’s a sign of, you know, a good lesson if people who don’t have to engage with it actually want to. I think that’s, that’s a sign of good activity.

He also saw the relevance to other subjects, such as problem-solving in maths, applied maths and physics: ‘Being able to break down, identify the problem and break it down into stages and then come up and see a pattern and to use a strategy is very relevant in those subjects’ (Teacher 11)

8.5.3.3 Student Learning

Associated with the course effectiveness are student learning and student outcomes. In the semi-summative phase, student learning was assessed in three ways: 1) Likert items (on the End of Course questionnaire) 2) an open-ended question on Learning (on the End of Course questionnaire) and 3) pre and post MCQ tests. Student Learning outcomes were concerned with the impact the course had on students, and was linked with student learning and awareness of Computational Thinking.

8.5.3.3.1 Quantitative Results: End of Course Questionnaire

The End of Course questionnaire contained nine items related to learning

- I am dissatisfied with the activities used in this course (guskey_learning2_reverse)
- I engaged in activities that helped me learn ideas or skills that were new and unfamiliar to me. (m_activation1)
- In this course, I was able to connect the activities to new ideas and skills I was learning. (m_activation3)
- The activities used in this course were helpful in learning.(unplugged_1)
- The activities helped increased my knowledge and skills in Computational Thinking. (unplugged_2)
- I see how I can apply what I learned in this course to other subjects. (application_5)
- I see how I can apply what I learned in this course to Computer Science. (application_6)
- Compared to what I knew before I took this course, I learned a lot. (satisfaction_1)
- I did not learn a lot in this course. (satisfaction_2_R) (reversed)

The mode and median values for each of the eight items per school are shown in Figures 8-10 and 8-11. Note items application_5 and application_6 were introduced new for this

phase of the course to ascertain students' perceptions on how they could use their new problem-solving skills and knowledge.

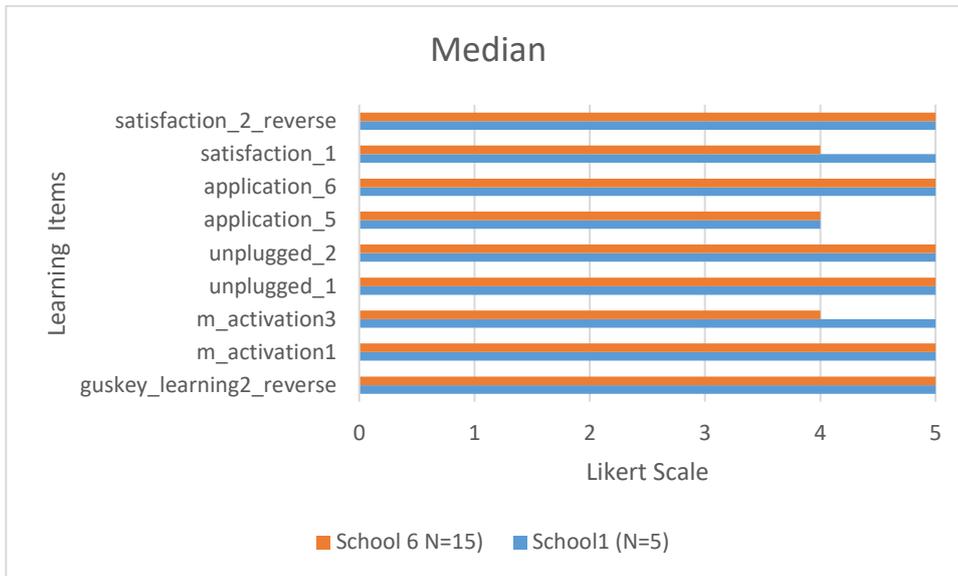


Figure 8-10 Median Values from School 1 (V3) and School 6 showing items related to learning

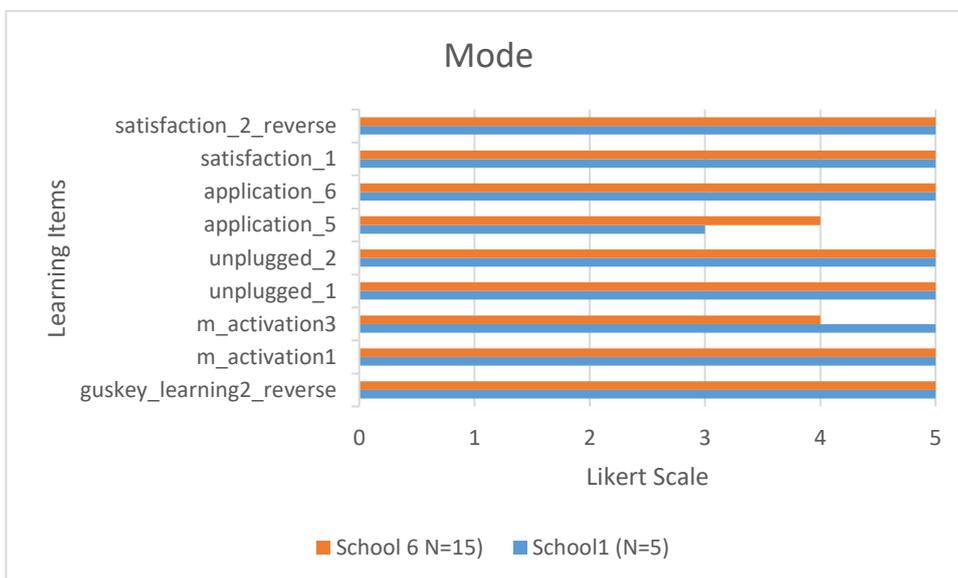


Figure 8-11 Mode Values from School 1 (V3) and School 6 showing items related to learning

What stands out in Figures 8-10 and 8-11, is that the mode and the median value for each item (bar 1) is either 4 or 5. This corresponds to Agree or Strongly Agree. It is apparent that the majority of students found the activities helped them learn new skills, and increase their knowledge of Computational Thinking. They also strongly agree that they learned a

lot in the course, and saw their learning progress. These results correspond to the positive findings from the corresponding Likert items in Version 1 of the course, which all had a mode and median value of 4 and 5 (see 6.6.1.1.2.3 and 6.6.1.4.1.). A notable point regarding the results from the semi-summative phase is that 80% of students in School 6 and 60% of students in School 1(V3) stated that they could apply what they learned to other subjects. This was coupled with 86% of students in School 6 and 100% of students in School 1 (V3) stating that they could apply what they learned to Computer Science.

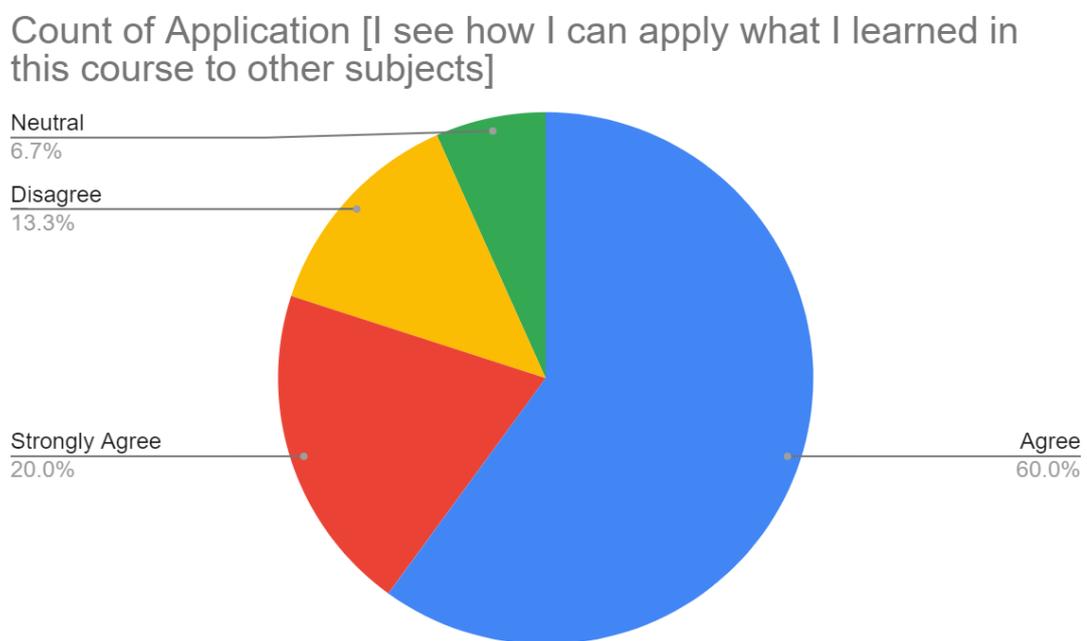


Figure 8-12 Pie chart showing results from School 6 to the statement: *I see how I can apply what I learned in this course to other subjects* (N=15)

8.5.3.3.2 Qualitative Data: Student Learning

Students' learning was further explored with the open-ended qualitative question: What do you think you learned during the course? Similar to before, each school's findings are tabulated before they are discussed as part of the analysis. Students gave a wide range of answers that were varied in detail, with some being quite informative. These answers were submitted online, which may be why they are not as detailed as the answers from Version

I submitted on paper. A notable point is five students mentioned the ‘intelligence’ of the computers, with four students correctly stating that computers are not smart, which was a learning objective of the course.

Another notable going forward point is to highlight to students the importance of clarity of language. This answer by a student reflects one of the learning goals of Unit 2, to highlight that computers follow algorithms, they are not smart: ‘That computers aren’t intelligent and rely on humans to write programmes them with data (they can only read programme) and that computers can be biased’ (Student G7). However, whilst the answer correctly states that humans write computer programmes, the student also states that a computer can be biased. This answer highlights the importance of language clarity. Computers are not biased but the data they use or algorithms they run maybe. An observation by this researcher is that, going forward, I would include explanations of the word thinking and intelligence in the course. Notably, no students stated that they learned how to think like a computer: ‘I learned about the thinking that goes on while developing computer programmes’ (Student G3). In this phase, I was always conscious of my language, as reflection on my teaching in Prototype phase V1, highlighted how I spoke about ‘computers thinking in two’ in the last unit. Table 8-5 and Table 8-6 display a sample of the students answers categorised and the frequency of the category.

Table 8-5 Results from the analysis of the open-ended question: What do you think you learned during the course? (N=15, School 6)

Category	Description	Example	Frequency Column	
			Mentions	By Person
Logic	Answers mentions logical thinking	I learned a lot about logical thinking, problem solving and programming (and also a few cool card tricks/riddles) I learnt how to solve problems using skills from Computer	4	4

		Science, and how to think logically to solve problems		
AI/Ethics	Answers mention AI or Ethics	i learned about computers and their artificial intelligence and what computational thinking actually is i also learned that computers are smart What artificial intelligence is and what type of searches there are.	3	3
Computational Thinking	Answers referred to CT	I learned about the thinking that goes on while developing computer programmes What computational thinking is. How it works. How it's applied. And how to write codes	4	4
Computer Science	Answers mention Computer Science	'I learned interesting information about computational thinking, ethical thinking and computer science that have opened my eyes to different ideas that I had never considered before'.	1	1
Programming	Answers relate to programming	'the basics of programming and the reasoning behind it all' 'I learned that computers aren't exactly intelligent and it is up to coders to be extremely specific with their commands.' 'I learned about the thinking that goes on while developing computer programmes'	4	4
Computers	Answers relate to computers	'That computers aren't intelligent and rely on humans to write programmes them with data (they can only read programme) and that computers can be biased' 'During this course I learned that Computers aren't intelligent and need the input of humans in order to work'	3	3

Table 8-6 Results from the analysis of the open-ended question: What do you think you learned during the course? (N=5, School 1(V3))

Category	Description	Example	Frequency Column	
			Mentions	By Person
Coding	Answers mention coding	‘I learned how to do python code and have a new appreciation for the course’ ‘I was able to remember a lot of what I already knew of pseudocode and it was really enjoyable to fully understand the terms involved with this’	2	2
Computers	Answers mention Computers	‘I learned that computers aren't as smart as we think and that if we were to go to war with them we would easily win’	1	1
Computer Science	Answers mention Computer Science	I learned the basics of computer science and how it affects everyday life.	1	1
Computational Thinking	Answers mention Computational Thinking	‘Computational Thinking’	1	1

8.5.3.3.3 Pre and post tests: MCQ

Finally, pre and post tests were issued to students at the start and end of the course. The tests consisted of twenty-one questions. They assessed topics from all units such as Computational Thinking terms, algorithms, binary and linear search, ethics and programming fundamentals. The tests are available in Appendix E.

In School 6, thirteen students completed the pre test and fourteen the post test. I was able to match twelve tests, as one number did not match. Using SPSS, a box plot of the mean values pre and post test were calculated and shown on the same chart see Figure 8-13

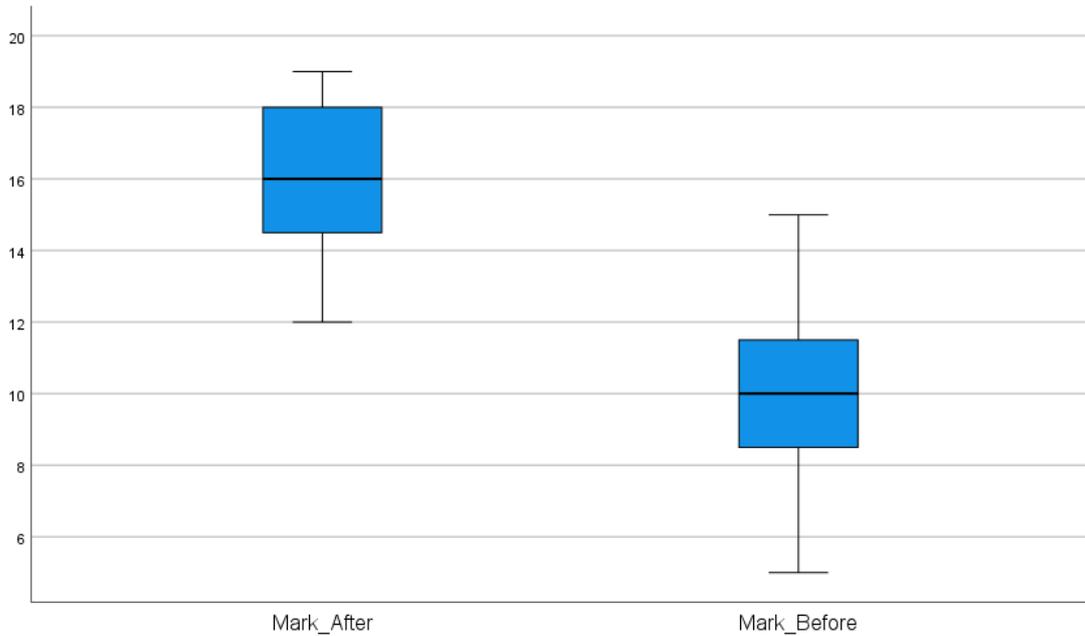


Figure 8-13 Box Plot of Mean scores pre and post course for School 6

From the box plot, it can be observed that the centre of the ‘Mark_After’ results is higher than the ‘Mark_before’ result. Using SPSS a paired sample t test was run to compare the means of the before and after scores. The result shows a significant average difference between the before and after results as, the average difference between the results is 5.83333 with the probability value (p) = 0.000099. Thus $p < 0.001$. To be significant, the p value should be smaller than or equal to 0.05. $t(12) = 5.932, p < 0.001$

Paired Samples Test									
		Paired Differences				t	df	Sig. (2-tailed)	
		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
					Lower	Upper			
Pair 1	Mark_After - Mark_Before	5.83333	3.40677	.98345	3.66878	7.99789	5.932	11	0.000099

Figure 8-14 Paired Sample t Test for School 6

This is a very promising finding, especially when one considers that students did not study for this test, had no coursebook or handouts, and received no homework. In addition, there is a marked increase in students’ correct answers regarding coding, algorithms, and artificial intelligence pre and post test.

Regarding the five students in school 1, unfortunately, a t-test could not be run on their data. I was unable to match the pre-and post-test results, and I also had one student who did the pre-test and one who did not do the post-test and vice versa. However, there was a 26% increase in their post test mean results.

8.5.3.3.4 Summary

Two Computational Thinking experts reviewed the teacher's guideline book for Version 3 (see 8.5.4). Both experts agreed that the course had the potential for developing students' Computational Thinking. Expert 4 provided the following rationale: 'you've come from this kind of very rich lovely history that we have of outreach work with all these gorgeous activities which do ignite and motivate people ...'. Expert 3 concurred: 'I think your curriculum definitely has the potential to develop their understanding of CT'. But he offered the following caveat: 'Now obviously measuring that is a very difficult task, and quantifying it and it's something that we found, when we developed ours, was something really difficult ...'. During this course, students' learning was assessed by various means and followed an exploratory path. In Version 1 and Version 3, students were asked if they felt they learned new skills and knowledge, the majority of students overwhelmingly agreed that they did. Students' knowledge was also evaluated using open questions and MCQ tests. The findings showed an increase in their knowledge, with the results from School 6 being statistically significant. In School 2 (V2), students' knowledge was assessed using a project, with Teacher 2 reporting high grades from the students' work. However, students were not assessed on how good they were at Computational Thinking, just on their understanding of concepts and theoretical knowledge. The overall course findings suggest that students gained an awareness of Computational Thinking, and in doing so, some students acquired a language to articulate their thinking. This was especially shown in the open-ended Q3 question and post-test answers in Version 1.

Students were also introduced to some core concepts of Computer Science. For example, in School 6, students' answers to test questions show improved scores (see Table 8-7).

The exploratory path of assessment is important to mention, as learning goals were confirmed and added as the study progressed. Two such goals are: 1) the importance of learning Computational Thinking to understand how computers work. In Version 2, teachers reported students as believing in a literal interpretation of '*deus ex machina*' (god from the machine). The second goal is for students to learn Computational Thinking to gain a language to articulate thinking. This was reported as aiding the transferability of CT knowledge. Lu and Fletcher (2009) previously argued for a Computational Thinking Language (CTL) in the absence of programming to teach the vocabulary and symbols to describe computation and abstraction. I argue for a CTL language that helps students articulate their thinking process, not as a precursor to programming, but one that can be related to all disciplines.

Table 8-7 A snapshot of the pre and post MCQ test scores for School 6

Question	Pre (right answers)	Post (right answers)
2 Decomposition	8	13
4 Computers	4	14
6 abstraction	7	12
7 Algorithmic thinking	6	13
8 Algorithms	4	10
12 Linear Search	8	14
13 Binary Search	5	15
15 AI is	1	12
20 Code example	8	15
21 Code example	6	10

8.5.3.4 Teacher Learning

This theme is concerned with one of Guskey's four levels of data to ascertain the effectiveness of a course, participant learning. To help to ascertain this information Guskey (2002) poses the question: 'Did participants acquire the required learning and skills?' For

the purpose of this phase, the question is: ‘Did teachers acquire the knowledge to teach this course?’ and also ‘How will teachers apply their new knowledge and skills gained from the course?’ Teacher 11 stated that he ‘would be very confident to teach this course. Like his students, he also stated that he felt he had developed or been re-introduced to new skills for problem-solving.

I felt that they did develop strategies for breaking down the problem and looking for patterns. And that maybe aren’t so evident when you are kind of stuck within say a maths class or, you know, having taught maths for so long, we never kind of pull on those skills. Yes, we may say - well what’s the pattern but like reducing the noise or avoiding the noise and stuff like that, we never really kind of take on those aspects which is a pity. And which, well what I’ll be doing in my teaching is, I will definitely be using some of those skills in it. Now, I thoroughly enjoyed it and I felt I learned a lot from it, yeah. (Teacher 11)

Teacher 11 also stated that he: ‘learned some things that I can incorporate within my other teaching but also maybe for, as an introduction or a support or foundation for the bit of coding that I do’. He was inspired by Unit 5, where the unit started with the house (finished code) instead of the bricks (programming fundamentals), and then got students to look for the bricks. He also liked that I used an online compiler, impressed upon students that there is a learning curve with programming while still managing to get them to experience success.

What was evident from Teacher 11’s interview was that he saw the course first as a problem-solving course and not as a pathway to programming.

But it’s, it’s that idea of not just coding for coding’s sake but that it fits into something bigger and for the student’s kind of gain and what they benefit from. They may never write another piece of code in their lives but if they are being able to analyse a problem, look for the pattern, come up with a strategy, that’s very, very powerful stuff, as I say even if they never write a single piece of code again. (Teacher 11)

He saw the course as being applicable outside of Computer Science. He stated that with problem-solving, it is the journey that is important

But it's not always getting that final answer, it's the process it's going through. And for students who are not high flyers, that's important, you know, they may never, you know, become coders or, you know, be able to be really good at maths or physics but that they can, they have an appreciation of what's involved and they can do some part of it and that their confidence is built by that. (Teacher 11)

He also alluded to the fact, that: 'problem-solving skills that can be taken from computational thinking, transfer very easily into other, those other subjects, those other STEM subjects really' (Teacher 11).

His comments share parallels with Teacher 7 in Version 2, who saw the value of the course, as way of informing students how computers work, giving them a base. She articulated that those who have an interest in computers will find their way: 'But it's the mass of the others... it's the guy who's going to be a farmer eventually or going to be whatever, who will rely on technology, they're the ones... that's why I think it's great to give them all a base, because the other guys will find their way anyway.'

8.5.4 **Quality**

Quality was defined as having three criteria: validity, practicality, and effectiveness (Nieveen, 1999a). The effectiveness of the course was discussed in 8.5.3. With reference to the content validity, its content was reviewed previously by two external experts, and their recommendations and changes were implemented. Two experts also critiqued the teacher's coursebook in the semi-summative phase. As stated earlier, both experts agreed that the course could develop student understanding of CT. With reference to the question 'Does the content of the course include relevant subjects and topics?' Expert 3, strongly agreed stating that:

definitely. I think there's a really nice combination of topics in there. I think the... all the topics from the four pillars of CT that you've identified are being touched on throughout the lessons. ... So, a big 'Yes' to your question. I think it includes relevant subjects and topics, yeah. (Expert 3)

Expert 4 stated she could not answer the question, as she did not have all the required information at hand, such as prior learning situations and their relationship to the learning objectives. This study was both exploratory and context specific to Ireland, and thus further work should take into account specifying the results from the pre-tests to highlight the prior knowledge of Irish TY students in relation to CT for different audiences. This would also result in strengthening the learning objectives.

Expert 4 also highlighted the importance of clarity, stating the need for my course book to have a concept guide. This has now been rectified. She also discussed the importance of clarity of language. Of note, she stated that abstraction is about hiding or ignoring details, not about discarding detail, which is different to my original definition which used the word **remove** (NCCA, 2016, p. 27). I subsequently changed my definition (see 5.7.1.3).

Expert 4 also had concerns with the course associating the making of a jam sandwich with an algorithm. I have clarified my reasoning for this in this thesis. This reasoning will also be placed in the teacher coursebook. I play the video on making a jam sandwich to highlight the importance of exact instructions. It is a conceptual algorithm, and the video shows a Dad making a sandwich with no 'human judgment', he follows the instructions exactly (Darnit, 2017). In the teacher workshop, I explained to teachers that they should act like a computer when interpreting the students' follow up activity of writing an algorithm for drawing emoji (see 7.1.4). I failed to say this in the teacher coursebook. This has been rectified to avoid misunderstandings.

The second quality criterion is practicality. Findings from the prototype phase showed that the course was shown to work best in a standard classroom, with teachers suggesting that

they believed a 80 minute class would work better than a 40 minute class. This course was tried out with twenty different class forms from six schools. Feedback from both teachers and students was used to revise the delivery of the course and its contents. All teachers who participated in the course stated that they would teach it again, with some teachers requesting the revised teacher coursebook for the year 2020-2021.

The third quality criterion is effectiveness, which has already been discussed see 8.5.3. In summary, these collective findings support the good quality of the course

8.5.5 **Summary**

The collective evaluations of the course's engagement, low threshold, quality, effectiveness, and practicality support the findings that the course is an engaging, low threshold, good quality, effective, and practical course for teaching and learning of Computational Thinking. Both students and teachers had positive reactions to the course, and felt that it had an impact on their learning of Computational Thinking, and problem-solving. The course was shown to be both practical, and sustainable. It was found to be effective and also to be of good quality. However, these findings have limitations. The course is context-specific, exploratory in nature, and more work is needed to document the progression from prior learning to learning objectives to assessment.

9 Conclusion

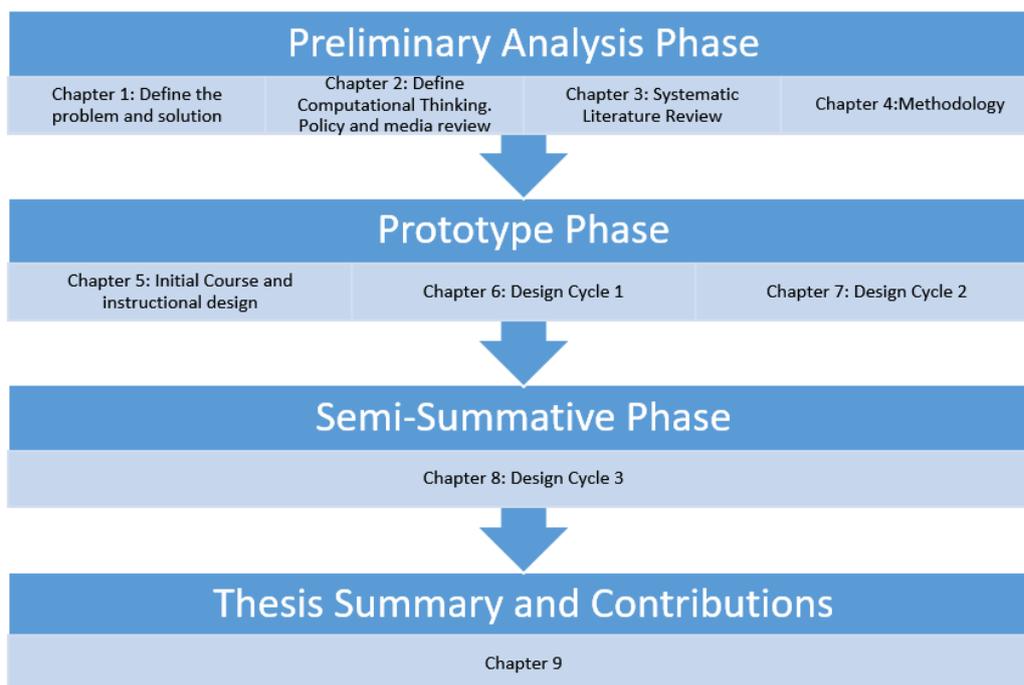


Figure 9-1 Thesis structure and outline

This final chapter starts with a summary of this PhD thesis, a thesis that resulted in two main practical contributions. The first of these was the development of a Computational Thinking course for Irish second-level students. The second was the ADAPTTER framework. These outcomes have implications for educational policy, initial teacher education, second-level curriculum, and second-level teaching practice in Ireland. These implications are discussed in this chapter. This PhD study also validated the use of unplugged activities as a teaching tool for Computational Thinking. Coupled with this validation, is the fact that detailed accounts of the EDR methodology and processes used in this PhD study are documented and illustrated in this thesis. This is important to note as current PhD students have been influenced by this study's systematic literature review process, its hybrid conjecture map, and the research approach illustrated in Figure 4-5. It is envisaged the future PhD students will be too.

This chapter also provides a summary of the key design components of the ADAPTTER framework. This framework was shown to result in a high quality, practical, engaging, effective, and low threshold course for the learning and teaching of Computational Thinking to Irish second-level teachers and students. This chapter concludes with a discussion on the dissemination and limitations of this PhD study and recommendations for future research.

This PhD study set out to answer the following question: **What are the characteristics of a high quality, practical, engaging, effective, and low threshold course for both the learning and teaching of Computational Thinking to Irish second-level teachers and students?**

It also set out to validate whether unplugged activities can be successfully used to teach Computational Thinking.

To answer these questions, the following work was undertaken. Chapter 1 introduced the research, specifically, the educational problem that this study addressed, how to teach Computational Thinking to Irish post-primary students, and its solution, a Computational Thinking course and the ADAPTTER framework.

Chapter 2 served to provide a working definition for Computational Thinking that considered the Irish position. This working definition resulted from a literature review of academic literature, educational documents, Irish educational policy and curriculum documents, and Irish newspapers. Findings from this review highlighted the importance of a universal definition of Computational Thinking, specifically in the Irish primary and post-primary curriculum.

Chapter 3 provided a systematic literature review on the following topics: Computational Thinking, online learning, post-primary students, and teaching/learning. This review served to investigate the role programming languages play in the teaching of Computational Thinking and also the current pedagogical approaches used to teach Computational Thinking online. This review helped identify the interventions, pedagogies, and materials that served as sources of inspiration for the curriculum content. It also highlighted the relative lack of literature related to unplugged activities online, which resulted in a further literature review of unplugged activities ‘offline’ in Chapter 5.

Chapter 4 outlined the philosophical and methodological framework that underpinned this study, pragmatism and EDR. The methodological methods and analytical techniques used were discussed in detail. This included the ‘Validation Analysis’ stage I developed to cater for the specific time constraints and school schedules that impacted this research.

Chapter 5 provided an overview of the initial content and design of the Computational Thinking course, and the rationale for the content and initial design. Chapter 5 also clarified what is understood by key terms in this study: practicality, engagement, quality, effectiveness, and low threshold. It also provided the results of a literature review on unplugged activities.

Chapter 6 concerned the piloting of Version 1 of the Computational Thinking course. It presented a detailed narrative on its implementation and evaluation of the content, quality, and engagement of the course. This version was piloted in four schools with eighty-six students and five teachers. The course content was also reviewed by two content experts.

Chapter 7 concerned the piloting of Version 2 of the Computational Thinking course. This pilot was concerned with the practicality and sustainability of the design and course. This version was piloted in two schools with 340 students and six teachers.

Chapter 8 concerned the semi-summative evaluation of Version 3 of the course. This version was piloted in two schools with twenty students. The course content was also reviewed by two content experts. This chapter provided the course's collective evaluations, which supported the findings that the course is an engaging, low threshold, good quality, effective, and practical course for the teaching and learning Computational Thinking.

9.1 Implications for Irish Educational Policy

This PhD study started with outlining an educational problem. In doing so, it specifically referred to the Department of Education and Skills' Digital Strategy report which called for submissions:

for the development of students' digital literacy by including coding and programming in the Irish primary and post-primary curriculum so that every learner has an opportunity to learn skills such as computational thinking, logic, critical thinking and strategic thinking to solve problems (DES, 2015, p. 22)

In a follow-up action plan 2018, the Department of Education and Skills refer to funding for a project that will use robotic materials to teach Computational Thinking (DES, 2018b). This PhD study has shown how unplugged activities can be successfully used to teach and learn Computational Thinking in an engaging, high quality, effective, and low threshold (including low cost) way to Irish students and teachers. A new digital strategy is currently being developed (DES, 2021). This PhD study recommends that unplugged activities be included in this strategy as a pedagogical approach to teaching Computational Thinking. This PhD study also recommends that its Computational Thinking course and ADAPTER framework be considered in this strategy. One of the findings from a report from UNESCO, the International Federation for Information Processing (IFIP), and the Technical Committee on Education on coding and Computer Science curriculums was to: 'Make increased use of "unplugged activities", which are typically less resource-intensive, to develop conceptual understanding in computer science' (Storte et al., 2019, p. 7).

My PhD study also recommends that the Department of Education and Skills provide a definitive definition of Computational Thinking in their new digital strategy. Coupled with the call for this definition is a call for clarity, it is recommended that the Department of Education and Skills clarifies the applicability of Computational Thinking outside of programming. Current Irish policy documents (DES, 2015; NCCA and Primary Developments, 2018) highlight the importance of Computational Thinking in connection to coding. The Leaving Certificate Computer Science curriculum document highlights the problem-solving framework of Computer Science outside of Computer Science (NCCA and DES, 2018). Components of Computational Thinking are also highlighted in the Senior Cycle Key Skills framework document (NCCA, 2009). This clarity is also applicable to the terms abstraction and algorithms in reference to Computational Thinking, as it has consequences for curriculum design. For instance, the Primary Maths background defines abstraction as ‘removing unnecessary detail’ (NCCA, 2016, p. 27). The detail is not removed, it is just ignored (Expert 4).

This PhD study argues that it is more important that all students learn how a computer works and how it is used to solve problems rather than the syntax and semantics of a programming language. Not all students are interested in learning to programme, but they all live in a digital world. Therefore, it would be beneficial to students to understand how computers work and are used to solve problems. Computational Thinking is, of course, important to programming, but this study recommends that unplugged activities be initially used to provide a conceptual model of Computational Thinking.

Findings from this study revealed how it is the language of Computational Thinking that facilitated the transfer of Computational Thinking concepts to other subjects. Teachers and students now had common words and terms to articulate their thinking, specifically in relation to logical thinking and problem-solving.

This study resulted in a fully developed Computational Thinking course. As part of its development, a workshop was run over one day to teach the course contents to teachers. This workshop/course can be used for future teacher education programmes at third level and also for teachers continuing professional development. The course can also be used to build confidence in teacher awareness and understanding of Computer Science and Computational Thinking.

9.2 Implications for Irish Curriculum

The outcomes of this PhD Study (the Computational Thinking course and the ADAPTTER framework) have implications for the Irish educational curriculum. This PhD study recommends that students learn Computational Thinking before they learn to code, specifically before they undertake the Junior Cycle Short Course in Coding or programming as part of the Leaving Certificate Computer Science course. It is also recommended that all students have the option to study Computational Thinking. It is strongly advocated that this is provided as an option for Transition Year students. This PhD's Computational Thinking course can be used standalone as is, or the individual course units can be used and adapted by teachers for their specific needs.

Through the broad applicability of Computational Thinking, students are introduced to Computer Science, problem-solving, and programming. However, care must be taken to ensure elements of Computational Thinking are not overgeneralised (Denning and Tedre, 2019), and that students understand the 'no human judgement' aspects of algorithms. By introducing Computational Thinking using unplugged activities in a planned progression, the cognitive load of Computational Thinking can be reduced for beginners before students are introduced in a scaffolded way to the fundamentals of programming.

9.3 Implications for Irish Initial Teacher Education

The outcomes of this PhD Study (the Computational Thinking course and the ADAPTTER framework) have implications for Irish initial teacher education. This PhD study

highlighted the importance of a common articulation language when communicating the thinking process involved in problem-solving. Findings from this study highlight how teachers lack a common language to describe the process of thinking and as a rule, do not highlight logical thinking or abstraction when solving problems. It is hence recommended that teachers are explicitly taught the theory surrounding Computational Thinking.

Findings from this study showed that teachers lacked knowledge on what exactly Computational Thinking is.

Findings from Yadav *et al.*'s (2011) study explain the benefits of including Computational Thinking in teacher education. Their study showed that student teachers gained a more favourable attitude towards Computer Science and were more likely to incorporate computing principles into their teaching practice after learning Computational Thinking. This proved true in this PhD study also.

Insights from this PhD study are relevant and timely for the design of Irish Initial Teacher Education programmes, specifically as all programmes (primary and post-primary) must be realigned starting September 2022 with Céim (The Teaching Council, 2020a). One of the requirements of Céim is Digital Skills (The Teaching Council, 2020b). This study's CT course provides knowledge on how algorithms and programmes are developed, thus complementing and providing background knowledge to teachers' digital literacy. Esteve-Mon, Llopis and Adell-Segura (2020) highlight how Spanish Institutions have included Computational Thinking as a digital competence component for student teachers. They further state how their results confirm a correlation between Computational Thinking and digital competence, Computational Thinking and communicative literacy and Computational Thinking and technological literacy. Butler and Leahy (2021) state that exposure to CT should be a core element of initial teacher education. By incorporating Computational Thinking and its application via unplugged activities into Irish initial teacher education, the knowledge and skillset associated with Computational Thinking will

enhance Irish student teachers' digital competence. This study's CT course and ADAPTTER framework were designed, developed and trialled in the Irish second-level system with Irish second-level teachers. Thus the practical contributions from this course would directly benefit Irish post-primary initial teacher education. The Irish Draft Primary Curriculum Framework highlights how there are demands to include Computational Thinking in the curriculum (Primary Developments and NCCA, 2020). This study's CT course could be adapted for primary students, specifically those in fifth and sixth class.

9.4 ADAPTTER Framework and Guidelines

Prior to this study, a call was made for research to develop and evaluate 'teaching and learning pedagogies for unplugged approaches in primary and secondary schools' (Waite, 2017, p. 48). It was suggested that these studies should investigate 'how teachers are successfully embedding unplugged activities to teach computational thinking' (Waite, 2017, p. 48). This study contributes findings to answer these research objectives through the ADAPTTER framework.

Activities, **D**emonstrations, **A**pplication, **P**re-activation, **T**ransparency, **T**heory, **E**xemplification, and **R**eflection (ADAPTTER) are the characteristics of a high quality, practical, engaging, effective, and low threshold course for both the learning and teaching of Computational Thinking to Irish second-level teachers and students.

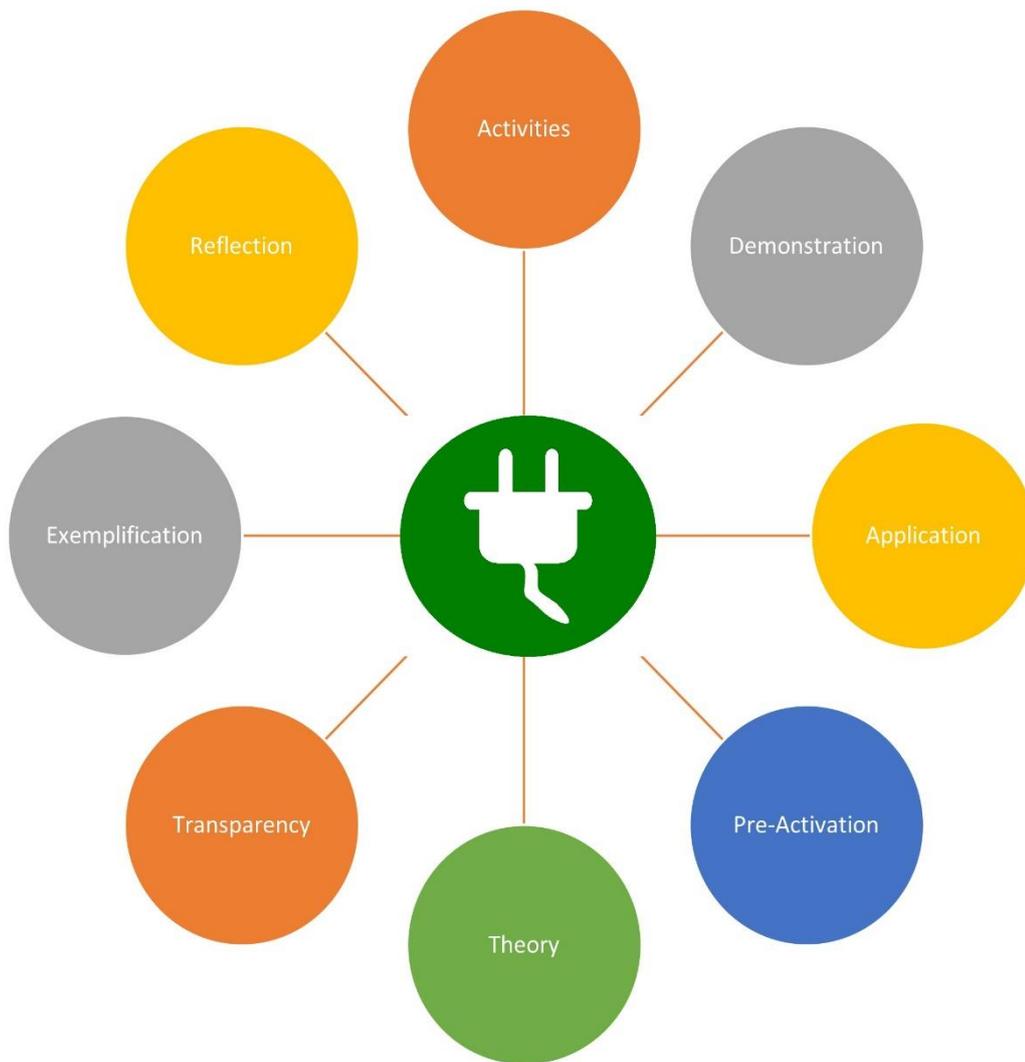


Figure 9-2 The ADAPTER framework for developing a high quality, effective, low threshold, engaging, and practical course for teaching Computational Thinking

Activities

This study shows that unplugged activities were an important component for the teaching and learning of Computational Thinking. These findings can be added to the relatively few empirical studies conducted on unplugged approaches in a regular classroom (Bell and Vahrenhold, 2018; Curzon *et al.*, 2019). For clarity, a distinction is made between activities that were ‘applied’ after a demonstration and activities that were used to establish a baseline of knowledge (see pre-activation).

With reference to activities used after a demonstration to apply new knowledge, this PhD study recommends that they be collaborative and related to either Computer Science or 'real-life'. Where appropriate, these group activities should be playfully competitive. The instructions for unplugged activities need to be clear, and a demonstration of concepts or usage of activities needs to precede the application of the unplugged activity.

(Demonstrations are discussed in the next section).

The activities that were used to establish a baseline of knowledge were characterised as being memorable. They took the form of a puzzle or games. These activities were not demonstrated first and they took place at the start of a lesson or new topic.

Demonstrations

Merrill's (2002) First Principles of Instruction were used to inform the initial design of the course. One of his design principles, 'demonstrations,' is an important design component of this PhD's Computational Thinking course. Merrill argues that too much instruction is 'tell rather than show' (Merrill, 2013, p. 23). This PhD study recommends that unplugged activities are placed in an instructional design that balances show with tell. Demonstrations promoted learning and engagement as they allowed students to approach a problem with a base of knowledge and clarity to the task at hand. This PhD study recommends the usage of short, relevant videos or websites to demonstrate concepts. For example, the 'Exact Instruction Challenge' video or websites such as 20Q.net or CodeMaster game at <http://csunplugged.mines.edu/codebreaker/game.cgi> were used to aid clarity to unplugged exercises. In the form of videos, demonstrations were also used to provide a hook to or to illustrate the truth and relevance of an activity. For example, movie trailers for 'Diving Bell and Butterfly' and the 'The Man Who Saved the World' were shown before their corresponding activities of 'Searching to Speak' and 'Ethics discussions'. This PhD study

recommends that a demonstration precede the application of knowledge using unplugged activities

Application

Similar to ‘Demonstrations’, ‘Application’ of knowledge is a principle of instruction that has its roots in Merrill (2002). It is recommended that students are given the opportunity to apply and practice the new knowledge demonstrated to them. This PhD study found that unplugged activities provided students with the opportunity to practice and apply their knowledge of Computational Thinking components. These activities used everyday items and took into consideration the variance of students’ Computer Science knowledge. This addresses two challenges reported by Computer Science teachers in the UK, technical problems and the ‘differentiation to meet different level of abilities’ (Sentance and Csizmadia, 2017, p. 479). Except for the programming activities used in this course, the activities used to apply knowledge took the form of real-world problems such as developing a spell checker, designing a protocol to help a person communicate, or developing a chatbot. The programming activities did not take this form, as students did not have the programming knowledge to facilitate this. This PhD study makes the recommendation that the application of knowledge includes peer collaboration.

Pre-activation

This design component has its foundations in Merrill’s ‘Activation’ principle of instruction, which states that ‘Learning is promoted when learners activate existing knowledge and skills as a foundation for new skills’ (Merrill, 2013, p. 21). This PhD study found that students did not have a common knowledge base connected to Computational Thinking concepts. This finding correlates with Izu *et al.*’s (2017) who also state that concepts central to Computational Thinking are alien to many K12 teachers. This PhD study recommends a pre-activation step, a short, simple, memorable, and fun activity

performed at the start of a lesson. The knowledge gained from the pre-activation activity is activated later in the lesson to build on more complex knowledge (before another activity is used to practice this knowledge). This study found that pre-activation activities facilitated teachers when transferring knowledge gained from unplugged activities to programming. The pre-activation activities always occurred in the same lesson as that where theoretical concepts were discussed and the activities related to said concept.

Transparency

This study found that transparency is an important design characteristic for content quality and engagement. The second chapter of this PhD thesis opened with a question: what is Computational Thinking? The lack of consensus regarding the scope and nature of Computational Thinking was discussed in detail in Chapter 2 (National Research Council, 2010). Coupled with this is the lack of clarity on definitions associated with algorithms and abstraction. Teachers and curriculum designers must state clearly their concept definitions. This PhD study found that clarity of language is important. Terms such as ‘thinking’, ‘sorting’, ‘intelligence’ are open to misconceptions when used with respect to computers. This study recommends that Computational Thinking courses clarify language, layout of course materials, concepts and instructional design. Gooder *et al.* (2012, p. 48) emphasize the ‘inseparable link between curriculum and instruction’. They highlight how the American ‘Exploring Computer Science’ course's instructional paradigm was as crucial as the selected content. This PhD study found this to be true in the Irish context also.

Theory

Unplugged activities were the main pedagogical tool used for teaching Computational Thinking, but this approach was supplemented with theory. Learning Computational Thinking concepts was shown to facilitate the transfer of this knowledge to other subjects. In their Computational Thinking course for Irish post-primary students using Python,

Mooney *et al.* (2014) state that students ‘found the material challenging, and both students and teachers reported that a less theoretical and more practical approach might have been more helpful for introducing key concepts’ (2014, p. 11). This study has helped address their recommendation by providing such a course combining practice and theory. It makes the furthermore specific recommendation that theoretical knowledge is introduced after the pre-activation step.

Exemplification

This design characteristic is intrinsically linked with demonstrations. Findings for this study suggest that it is not ‘good enough’ to have real-world examples, they should, where possible, be context-specific as well. Findings from this study highlighted students’ difficulties with real-world examples like ‘London Underground’ or the ‘Dublin Bus App’. Students were unable to relate to them and needed localised examples. This study recommends that exemplifications of concepts are relatable to students and that these concepts are ‘shown rather than told’. It is also recommended (once students have a baseline of knowledge) that students themselves contribute to providing Computational Thinking examples from their environment. This will ensure teachers have a database of relevant student context-specific examples.

Reflection

This design component has its foundations in Merrill’s integration principle (Merrill, 2002). Merrill states that ‘Learning is promoted when learners reflect on, discuss, and defend their newly acquired knowledge and skill’ (2013, p. 29). This PhD study recommends that students are given the opportunity to demonstrate, reflect and share their learning. Teacher preference should be taken into account concerning the type of reflections, be it verbal or written. It is also recommended that homework be assigned to students as a mechanism to provide individual feedback of their work.

9.5 Implications for Practice

The resources need to teach this course are low cost. The course can be conducted in a standard classroom (except for the last unit). This PhD study recommends that its Computational Thinking course is timetabled for 80 minutes classes and proceeds the study of programming. However, this course is flexible in how its units can be taught such that units 1, 2 and 5 can be completed together, and Unit 4 can be taught standalone. This PhD Computational Thinking course introduces students to Computational Thinking, Computer Science, problem-solving, and programming fundamentals. Not all students want to be programmers but they all live in a digital world.

9.6 Limitations

EDR defined the research approach, informed the research questions, impacted this study's outcome and influenced the structure of this thesis. In chapter 5, limitations associated with EDR were discussed in detail. The first limitation is concerned with the overwhelming amounts of data collected during EDR, which can result in data not being analysed due to time constraints (Brown, 1992). This proved true in this study. Due to time constraints, artefacts collected from this study were not all analysed. Also due to time constraints and difficulties encountered by teachers, only the MCQs from Version 3 of the study were analysed. However, these MCQs were triangulated with students' questionnaire data. The second limitation associated with EDR is that it is context-specific. This limitation is addressed in the 'Future Work' section in regards to generalisation (see 9.6).

The third limitation associated with EDR is researcher bias. To overcome this concern, I ensured that this thesis provided context-rich discussions on my settings, design decisions and research results. However, due to the amount of data I collected, coupled with the word-count limitation of this thesis, I did struggle in knowing what to include and more importantly, what to exclude.

Finally, a limitation of the findings from Version 3, is the small sample size of students (N=20). It is therefore recommended that future work involves this course being replicated by different teachers and researchers. This would also serve to overcome the third limitation.

9.7 Future Work

In EDR, generalisation is concerned with transferring the local instructional theories and practical intervention to other educational settings (McKenney and Reeves, 2012).

Replication supports generalisation. In this study, this course was replicated twenty-one times in six different schools. To support further replication, the contents of this course, including its course guide book, its slides and unplugged materials, will be made available online through my computationalthinking.ie website, and circulated via the CESI mailing list, and by social media. This study's practical intervention of a Computational Thinking course and ADAPTTER framework was designed for an Irish problem which should aid its transfer to other Irish post-primary schools. McKenny and Reeves (2012, p. 21) describe case-to-case generalisation as 'the transfer of ideas that takes place when a person in one setting considers adopting an intervention, or its underlying ideas, for use in another setting'. To support case-to-case generalisation to non-Irish settings, descriptions of the salient characteristics and context of this intervention are provided. It is ultimately the 'consumer' who will decide and make the transfer of this intervention to other contexts (McKenney and Reeves, 2012), and the rigour and detail that EDR studies require will aid this transfer.

Future proposed research could explore how the ADAPTTER framework could be applied to the teaching of Computer Science and Computational Thinking in the primary setting (NCCA, 2019). Teachers from Version 2 highlighted the importance of the workshop to their learning of CT. Future research could investigate turning the contents of the

workshop into MOOC to facilitate a wider dissemination of the course and translate the contents of the course to the Irish language.

9.8 Conclusion

As I write this last paragraph, I am conscious that my PhD journey is near its end. I set out to answer a question: **What are the characteristics of a high quality, practical, engaging, effective, and low threshold course for both the learning and teaching of Computational Thinking to Irish second-level teachers and students?** In doing so, I have made two practical contributions to knowledge. In making my course and framework available online, I hope that more teachers and students will benefit from these contributions. In documenting my systematic literature process and EDR approach, I hope that future PhD students can benefit from this work. Future researchers can use my ADAPTTER framework as is or as a springboard to a new framework. Finally, I hope that all Irish students have the option to learn Computational Thinking, and I would like to think I have helped a little with this.

References

- Ahmadi, N. and Jazayeri, M. (2014) ‘Analyzing the Learning Process in Online Educational Game Design: A Case Study’, in *Proceedings of the 2014 23rd Australian Software Engineering Conference*. Washington, DC, USA: IEEE Computer Society (ASWEC '14), pp. 84–93. doi: 10.1109/ASWEC.2014.34.
- Ahmadi, N., Jazayeri, M. and Landoni, M. (2012) ‘Helping Novice Programmers to Bootstrap in the Cloud: Incorporating Support for Computational Thinking into the Game Design Process’, in *Proceedings of the 2012 IEEE 12th International Conference on Advanced Learning Technologies*. Washington, DC, USA: IEEE Computer Society (ICALT '12), pp. 349–353. doi: 10.1109/ICALT.2012.24.
- Aho, A. V. (2011) ‘Ubiquity symposium: Computation and computational thinking’, *Ubiquity*, 2011(January). doi: 10.1145/1895419.1922682.
- Alexandron, G. *et al.* (2014) ‘Scenario-based programming: reducing the cognitive load, fostering abstract thinking’, in *Companion Proceedings of the 36th International Conference on Software Engineering - ICSE Companion 2014. Companion the 36th International Conference*, Hyderabad, India: ACM Press, pp. 311–320. doi: 10.1145/2591062.2591167.
- Anderson, T. and Shattuck, J. (2012) ‘Design-Based Research: A Decade of Progress in Education Research?’, *Educational Researcher*. doi: 10.3102/0013189X11428813.
- Angeli, C. *et al.* (2016) ‘A K-6 Computational Thinking Curriculum Framework: Implications for Teacher Knowledge’, *Journal of Educational Technology & Society*, 19(3), pp. 47–57.
- Axelson, R. D. and Flick, A. (2011) ‘Defining Student Engagement’, *Change: The magazine of higher learning*, February, pp. 38–43. Available at: <https://doi.org/10.1080/00091383.2011.533096> (Accessed: 4 August 2020).
- Bakker, A. (2018) *Design Research in Education: A Practical Guide for Early Career Researchers*. Oxford: Routledge.
- Bakker, A. and van Eerde, D. (2015) ‘An Introduction to Design-Based Research with an Example From Statistics Education’, in Bikner-Ahsbabs, A., Knipping, C., and Presmeg, N. (eds) *Approaches to Qualitative Research in Mathematics Education*. Dordrecht: Springer Netherlands (Advances in Mathematics Education), pp. 429–466. doi: 10.1007/978-94-017-9181-6_16.
- Barab, S. and Squire, K. (2004) ‘Design-Based Research: Putting a Stake in the Ground’, *Journal of the Learning Sciences*, 13(1), pp. 1–14. doi: 10.1207/s15327809jls1301_1.
- Barefoot CAS (2014) ‘Computational Thinking’. Available at: <http://barefootcas.org.uk/barefoot-primary-computing-resources/concepts/computational-thinking/> (Accessed: 28 June 2021).
- Barendsen, E. *et al.* (2015) ‘Concepts in K-9 computer science education’, in *Proceedings of the 2015 ITiCSE on working group reports*. ACM, pp. 85–116.

- Barr, V. and Stephenson, C. (2011) ‘Bringing computational thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community?’, *ACM Inroads*, 2(1), pp. 48–54.
- Basawapatna, A. (2016) ‘Alexander Meets Michotte: A Simulation Tool Based on Pattern Programming and Phenomenology.’, *Journal of Educational Technology & Society*, 19(1), pp. 277–291.
- Basawapatna, A. R. *et al.* (2013) ‘The zones of proximal flow: guiding students through a space of computational thinking skills and challenges’, in *Proceedings of the ninth annual international ACM conference on International computing education research*. ACM, pp. 67–74.
- Basawapatna, A. R. and Repenning, A. (2010) ‘Cyberspace Meets Brick and Mortar: An Investigation into How Students Engage in Peer to Peer Feedback Using Both Cyberlearning and Physical Infrastructures’, in *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education*. New York, NY, USA: ACM (ITiCSE ’10), pp. 184–188. doi: 10.1145/1822090.1822143.
- BBC (2014) *BBC - Ethics - Introduction to ethics: Ethics: a general introduction*. Available at: http://www.bbc.co.uk/ethics/introduction/intro_1.shtml (Accessed: 19 October 2019).
- BBC (2020) *What is computational thinking? - Introduction to computational thinking - KS3 Computer Science Revision, BBC Bitesize*. Available at: <https://www.bbc.co.uk/bitesize/guides/zp92mp3/revision/1> (Accessed: 2 July 2020).
- BBC (2021) *Abstraction - Fundamentals of algorithms - AQA - GCSE Computer Science Revision - AQA - BBC Bitesize, BiteSize*. Available at: <https://www.bbc.co.uk/bitesize/guides/zjddqhv/revision/3> (Accessed: 14 June 2021).
- Beecher, K. (2017) *Computational Thinking : A beginner’s guide to problem-solving and programming*. Swindon: British Computer Society.
- Bell, T. *et al.* (2009) ‘Computer Science Unplugged: school students doing real computing without computers’, *The New Zealand Journal of Applied Computing and Information Technology*, 13(1), pp. 20–29.
- Bell, T. and Lodi, M. (2019) ‘Constructing Computational Thinking Without Using Computers’, *Constructivist foundations, Vrije Universiteit Brussel*, 14(3), pp. 342–351. Available at: <https://hal.inria.fr/hal-02378761> (Accessed: 28 June 2021).
- Bell, T. and Vahrenhold, J. (2018) ‘CS Unplugged—How Is It Used, and Does It Work?’, in Böckenhauer, H.-J., Komm, D., and Unger, W. (eds) *Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday*. Cham: Springer International Publishing (Lecture Notes in Computer Science), pp. 497–521. doi: 10.1007/978-3-319-98355-4_29.
- Bell, T., Witten, I. H. and Fellows, M. (1998) ‘Computer Science Unplugged...off-line activities and games for all ages’, p. 240. Available at: <https://classic.csunplugged.org/wp-content/uploads/2015/01/unplugged-book-v1.pdf> (Accessed: 28 June 2021).

- Berkers, E. (2003) *Eliza - chat with this electronic therapist, Eclectic Energies*. Available at: <https://www.eclecticenergies.com/psyche/eliza> (Accessed: 16 May 2021).
- Biesta, G. J. (2010) 'Pragmatism and the philosophical foundations of mixed methods research.', in Teddlie, C. and Tashakkori, A. (eds) *SAGE handbook of mixed methods in social and behavioral research*,. 2nd edn, pp. 95–117.
- Bloom, B. S. (1956) 'Taxonomy of educational objectives. Vol. 1: Cognitive domain', *New York: McKay*, 20, p. 24.
- Bocconi, S. *et al.* (2016) 'Developing computational thinking in compulsory education– Implications for policy and practice; EUR 28295 EN'. Available at: http://publications.jrc.ec.europa.eu/repository/bitstream/JRC104188/jrc104188_computhinkreport.pdf.
- Boran, M. (2013) 'Technology', *The Irish Times*, 31 October, p. 4.
- Bowe, B. and Fitzmaurice, M. (2011) 'Guide to Writing Learning Outcomes'. Technological University Dublin. Available at: <https://www.dit.ie/media/ditltdc/documents/Microsoft%20Word%20-%20LearningOutcomesGuide.pdf>.
- Braun, V. and Clarke, V. (2006) 'Using thematic analysis in psychology', *Qualitative Research in Psychology*, 3(2), pp. 77–101. doi: 10.1191/1478088706qp063oa.
- Braun, V. and Clarke, V. (2012) 'Thematic analysis.', in Cooper, H. *et al.* (eds) *APA handbooks in psychology®. APA handbook of research methods in psychology, Vol. 2. Research designs: Quantitative, qualitative, neuropsychological, and biological*. <https://doi.org/10.1037/13620-004>,: American Psychological Association., pp. 57–71.
- Bremgartner, V., Netto, J. de M. and Menezes, C. (2017) 'Conceptual framework for collaborative educational resources adaptation in virtual learning environments', in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Wuhan, China, pp. 467–471. Available at: http://dx.doi.org/10.1007/978-3-319-61425-0_42 (Accessed: 28 June 2021).
- Brennan, K. and Resnick, M. (2012) 'New frameworks for studying and assessing the development of computational thinking', in *Proceedings of the 2012 annual meeting of the American Educational Research Association, Vancouver, Canada*, pp. 1–25.
- Brennan, K. and Resnick, M. (2013) 'Stories from the Scratch Community: Connecting with Ideas, Interests, and People', in *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*. New York, NY, USA: ACM (SIGCSE '13), pp. 463–464. doi: 10.1145/2445196.2445336.
- Briggs, D. (2017) 'BBC Four - The Secret Rules of Modern Living: Algorithms', *BBC*. BBC. Available at: <https://www.bbc.co.uk/programmes/p030s6b3> (Accessed: 10 July 2020).
- Brown, A. L. (1992) 'Design Experiments: Theoretical and Methodological Challenges in Creating Complex Interventions in Classroom Settings', *The Journal of the Learning Sciences*, 2(2), pp. 141–178. Available at: <http://www.jstor.org/stable/1466837> (Accessed: 18 June 2019).

- Brown, J. D. (2000) 'What issues affect Likert-scale questionnaire formats?', *Shiken: JALT Testing & Evaluation SIG Newsletter*, 4(1), pp. 27–30. Available at: http://hosted.jalt.org/test/bro_7.htm (Accessed: 11 June 2020).
- Brownlee, M. (2019) *Riding in a Driverless Taxi at CES 2019!* YouTube. Available at: <https://www.youtube.com/watch?v=gfWjsKsEry0> (Accessed: 22 July 2020).
- Bryman, A. (2012) *Social research methods*. 4th edn. Oxford: Oxford University Press.
- Buitrago Flórez, F. *et al.* (2017) 'Changing a Generation's Way of Thinking: Teaching Computational Thinking Through Programming', *Review of Educational Research*, 87(4), pp. 834–860. doi: 10.3102/0034654317710096.
- Butler, D. and Leahy, M. (2021) 'Developing preservice teachers' understanding of computational thinking: A constructionist approach', *British Journal of Educational Technology*, 52(3), pp. 1060–1077. doi: 10.1111/bjet.13090.
- Butterfield, A. B., Ngondi, G. E. N. E. and Kerr, A. K. (2016) 'sorting', in Butterfield, A., Ngondi, G. E., and Kerr, A. (eds) *A Dictionary of Computer Science*. Oxford University Press. Available at: <https://www.oxfordreference.com/view/10.1093/acref/9780199688975.001.0001/acref-9780199688975-e-4933> (Accessed: 5 February 2021).
- Byrne, J., Kearney, S. and Sullivan, K. (2018) 'Technology-Mediated Collaborative Learning: The Bridge21 Activity Model in Theory and Practice.', in Daniela, L. (ed.) *Didactics of Smart Pedagogy. Smart Pedagogy for Technology Enhanced Learning*. Cham, Switzerland: Springer, pp. 309–330. Available at: <https://doi.org/10.1007/978-3-030-01551-0> (Accessed: 14 July 2020).
- Caeli, E. N. and Yadav, A. (2020) 'Unplugged Approaches to Computational Thinking: a Historical Perspective', *TechTrends*, 64(1), pp. 29–36. doi: 10.1007/s11528-019-00410-5.
- Cambridge Dictionary (2021) *mindset*. Available at: <https://dictionary.cambridge.org/dictionary/english/mindset> (Accessed: 1 June 2021).
- Camp, T. and Rader, C. (no date) *CS Unplugged for Middle Schools*. Available at: <http://csunplugged.mines.edu/activity-codebreaker.html> (Accessed: 15 July 2020).
- Carini, R. M., Kuh, G. D. and Klein, S. P. (2006) 'Student Engagement and Student Learning: Testing the Linkages*', *Research in Higher Education*, 47(1), pp. 1–32. doi: 10.1007/s11162-005-8150-9.
- Cendrowski, M. (2009) *The Big Bang Theory - The Friendship Algorithm*. YouTube: CBS. Available at: <https://www.youtube.com/watch?v=k0xgjUhEG3U&t=43s> (Accessed: 11 June 2021).
- Chalaye, C. and Male, D. (2011) 'Applying Vygotsky's zone of proximal development and peer collaboration to pupils with profound and multiple learning difficulties and severe learning difficulties: two case studies', *The SLD Experience*, 61(1), pp. 13–18.
- Chew, C. and Eysenbach, G. (2010) 'Pandemics in the Age of Twitter: Content Analysis of Tweets during the 2009 H1N1 Outbreak', *PLOS ONE*, 5(11), p. e14118. doi: 10.1371/journal.pone.0014118.

- Cohen, L., Manion, L. and Morrison, K. (2011) *Research Methods in Education*. 7th edition. London ; New York: Routledge.
- Collins, A. (1990) *Toward a Design Science of Education. Technical Report No. 1*. Report. New York: Centre for Technology in Education, pp. 1–9. Available at: <https://eric.ed.gov/?id=ED326179> (Accessed: 18 June 2019).
- Computing at School (CAS) (2015) ‘Computing at School. Computational thinking: A guide for teachers. [Online].’ Available at: <http://community.computingatschool.org.uk/files/6695/original.pdf> (Accessed: 28 June 2021).
- Connolly, C., Murphy, E. and Moore, S. (2009) ‘Programming Anxiety Amongst Computing Students—A Key in the Retention Debate?’, *IEEE Transactions on Education*, 52(1), pp. 52–56. doi: 10.1109/TE.2008.917193.
- Creswell, J. W. (2014) *Research design: Qualitative, quantitative, and mixed methods approaches*. 4th edition. Los Angeles: Sage Publications.
- Creswell, J. W. and Plano Clark, V. (2011) *Designing and Conducting Mixed Methods Research*. 2nd edn. Los Angeles: Sage Publications.
- Croasmun, J. T. and Ostrom, L. (2011) ‘Using Likert-Type Scales in the Social Sciences’, *Journal of Adult Education*, 40(1), pp. 19–22.
- Csizmadia, A. *et al.* (2015) *Computational thinking A guide for teachers*. Computing At School. Available at: https://eprints.soton.ac.uk/424545/1/150818_Computational_Thinking_1_.pdf (Accessed: 28 June 2021).
- Curtin, M. (2018) ‘The 10 Most Commonly Misspelled Words in the English Language | Inc.com’, *Inc.*, 24 October. Available at: <https://www.inc.com/melanie-curtin/the-10-most-commonly-misspelled-words-in-english-language.html> (Accessed: 14 July 2020).
- Curzon, P. (2014a) *Computational Thinking Searching To Speak*. Queen Mary University of London. Available at: <http://www.cs4fn.org/computationalthinking/booklets/ComputationalThinkingSearchingToSpeak.pdf>.
- Curzon, P. *et al.* (2014) ‘Introducing teachers to computational thinking using unplugged storytelling’, in *Proceedings of the 9th Workshop in Primary and Secondary Computing Education. WiPSCE '14*, ACM Press, pp. 89–92. doi: 10.1145/2670757.2670767.
- Curzon, P. (2014b) ‘Unplugged Computational Thinking for Fun’, in Brinda, T. *et al.* (eds) *Commentarii informaticae didacticae 7. KEYCIT 2014- Key Competencies in Informatics and ICT*, Potsdam, Germany: Universitätsverlag Potsdam, pp. 15–28.
- Curzon, P. *et al.* (2019) ‘Computational Thinking’, in *The Cambridge Handbook of Computing Education Research*. Cambridge University Press, pp. 513–546.
- Curzon, P. and McOwan, P. W. (2017) *The Power of Computational Thinking*. London: World Scientific Publishing Europe.

- Dagiene, V. and Stupuriene, G. (2016) 'Bebras – a Sustainable Community Building Model for the Concept Based Learning of Informatics and Computational Thinking.', *Informatics in Education*, 15(1), pp. 25–44.
- Darnit, J. (2017) *Exact Instructions Challenge - THIS is why my kids hate me.* / Josh Darnit. YouTube. Available at: https://www.youtube.com/watch?v=cDA3_5982h8&t=36s (Accessed: 15 July 2020).
- Dasgupta, S. *et al.* (2016) 'Remixing As a Pathway to Computational Thinking', in *Proceedings of the 19th ACM Conference on Computer-Supported Cooperative Work & Social Computing*. New York, NY, USA: ACM (CSCW '16), pp. 1438–1449. doi: 10.1145/2818048.2819984.
- Davidson, S. and Murphy, C. (2018) *Computational Thinking for Problem Solving, 1.1 Introduction - Pillars of Computational Thinking*, Coursera. Available at: <https://www.coursera.org/lecture/computational-thinking-problem-solving/1-1-introduction-4twR7> (Accessed: 30 June 2020).
- Day, R. A. (1995) *How to write & publish a scientific paper*. 4. ed. Cambridge: Cambridge Univ. Press.
- Denning, P. J. (2017) 'Remaining trouble spots with computational thinking', *Communications of the ACM*, 60(6), pp. 33–39. doi: 10.1145/2998438.
- Denning, P. J. and Tedre, M. (2019) *Computational Thinking*. Cambridge Massachusetts: MIT Press.
- Department of Children and Youth Affairs (DCYA) (ed.) (2012) *Guidance for developing ethical research projects involving children*. Dublin: Government Publications. Available at: <https://www.lenus.ie/bitstream/handle/10147/311115/xEthicsGuidance.pdf> (Accessed: 28 June 2021).
- DES (2015) *Digital Strategy for Schools 2015-2020: Enhancing Teaching, Learning and Assessment*. Dublin: The Stationery Office. Available at: <https://www.education.ie/en/Publications/Policy-Reports/Digital-Strategy-for-Schools-2015-2020.pdf> (Accessed: 28 June 2021).
- DES (2018a) *Annual Statistical Report: Statistical Report 2016-2017*. Available at: <https://www.education.ie/en/Publications/Statistics/Statistical-Reports/Statistical-Report-2016-2017.xlsx> (Accessed: 28 June 2021).
- DES (2018b) *Digital Strategy for Schools 2015-2020 Action Plan 2018*. Available at: <https://www.education.ie/en/Publications/Policy-Reports/digital-strategy-action-plan-2018.pdf> (Accessed: 30 September 2019).
- DES (2019) *Annual Statistical Report: Statistical Report 2017-2018*. Available at: <https://www.education.ie/en/Publications/Statistics/annual-stistical-reports/statistical-report-2017-2018.xlsx>.
- DES (2021) *05 April, 2021 - Minister Foley announces development of a New Digital Strategy for Schools, Department of Education and Skills*. Available at: <https://www.education.ie/en/Press-Events/Press-Releases/2021-press-releases/PR21-04-05.html> (Accessed: 28 June 2021).

- Dewey, J. (1910) *How We Think*. Reprint 1997. New York: Dover Publications.
- Dewey, J. (1911) *The Middle Works 1899-1924 Volume 6*. Reprint 1978. Edited by J. A. Boydston. Southern Illinois University Press.
- Dewey, J. (1929a) *Experience and Nature*. London, United Kingdom: George Allen & Unwin Ltd.
- Dewey, J. (1929b) *The Later Works, 1925-1953 The Quest for Certainty*. Reprint 1988. Edited by J. A. Boydston. Southern Illinois University Press.
- Dewey, J. (1938a) *Experience And Education*. The 60th Anniversary Edition, 1998. Indiana, USA: Kappa Delta Pi.
- Dewey, J. (1938b) *Logic. The Theory of Inquiry*. Reprint 1939. New York: Henry Holt.
- Dhatsuwan, A. and Precharattana, M. (2016) 'BLOCKYLAND', *Simul. Gaming*, 47(4), pp. 445–464. doi: 10.1177/1046878116643468.
- Dixon-Woods, M. *et al.* (2006) 'Conducting a critical interpretive synthesis of the literature on access to healthcare by vulnerable groups', *BMC Medical Research Methodology*, 6, p. 35. doi: 10.1186/1471-2288-6-35.
- Dorph, R., Cannady, M. A. and Schunn, C. (2016) 'How Science Learning Activation Enables Success for Youth in Science Learning.', *Electronic Journal of Science Education*, 20(8), pp. 49–85.
- EPPI-Centre (2003) 'Review Guidelines for Extracting Data and Quality Assessing Primary Studies in Educational Research. Version 0.9.7.' London: EPPI-Centre, Social Science Research Unit. Available at: <https://eppi.ioe.ac.uk/cms/Default.aspx?tabid=184#Guidelines> (Accessed: 16 June 2018).
- EPPI-Centre (2006) 'EPPI-Centre methods for conducting systematic reviews'. London: EPPI-Centre, Social Science Research Unit, Institute of Education, University of London.
- Erdogan, M. *et al.* (2010) 'A Qualitative Study on Classroom Management and Classroom Discipline Problems, Reasons, and Solutions: A Case of Information Technologies Class.', *Educational Sciences: Theory and Practice*, 10(2), pp. 881–891.
- Escherle, N. A. *et al.* (2016) 'Piloting Computer Science Education Week in Mexico', in *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*. New York, NY, USA: ACM (SIGCSE '16), pp. 431–436. doi: 10.1145/2839509.2844598.
- Esteve-Mon, F., Llopis, M. and Adell-Segura, J. (2020) 'Digital Competence and Computational Thinking of Student Teachers', *International Journal of Emerging Technologies in Learning (iJET)*, 15(2), pp. 29–41. Available at: <https://www.learntechlib.org/p/217159/> (Accessed: 30 August 2021).
- European Commission High level Expert Group on Artificial Intelligence (2019) *A definition of Artificial Intelligence: main capabilities and scientific disciplines*. Text. Available at: https://ec.europa.eu/newsroom/dae/document.cfm?doc_id=56341 (Accessed: 20 October 2019).

- Faggella, D. (2018) *What is Artificial Intelligence? An Informed Definition | Emerj*. Available at: <https://emerj.com/ai-glossary-terms/what-is-artificial-intelligence-an-informed-definition/> (Accessed: 20 July 2020).
- Fauzan, A., Plomp, Tjeerd and Gravemeijer, K. (2013) 'The development of an RME-based geometry course for Indonesian primary schools', in Plomp, Tj and Nieveen, N. (eds) *Educational design research Part B: Illustrative cases*. Enschede,: SLO • Netherlands institute for curriculum development, pp. 159–178.
- Fereday, J. and Muir-Cochrane, E. (2006) 'Demonstrating Rigor Using Thematic Analysis: A Hybrid Approach of Inductive and Deductive Coding and Theme Development', *International Journal of Qualitative Methods*, 5(1), pp. 80–92. doi: 10.1177/160940690600500107.
- Fleming, J. (2018) 'Recognizing and resolving the challenges of being an insider researcher in work-integrated learning', *International Journal of Work-Integrated Learning*, 19(3), pp. 311–320.
- Fredricks, J. *et al.* (2011) *Measuring student engagement in upper elementary through high school: a description of 21 instruments*. Issues&Answers Report REL 2011– No. 098. Washington DC: U.S. Department of Education, Institute of Education Sciences, National Center for Education Evaluation and Regional Assistance, Regional Educational Laboratory Southeast., p. 88. Available at: <https://files.eric.ed.gov/fulltext/ED514996.pdf> (Accessed: 28 June 2021).
- Fredricks, J. A., Blumenfeld, P. C. and Paris, A. H. (2004) 'School Engagement: Potential of the Concept, State of the Evidence', *Review of Educational Research*, 74(1), pp. 59–109. doi: 10.3102/00346543074001059.
- Frick, T. W. *et al.* (2009) 'College student perceptions of teaching and learning quality', *Educational Technology Research and Development*, 57(5), pp. 705–720. doi: 10.1007/s11423-007-9079-9.
- Fry, H. (2018) *Hello World: How to be Human in the Age of the Machine*. London: Penguin Random House.
- Furber, S. (2012) 'Shut down or restart? The way forward for computing in UK schools', *The Royal Society, London*.
- Futschek, G. (2006) 'Algorithmic thinking: the key for understanding computer science', in *International conference on informatics in secondary schools-evolution and perspectives*. Springer, pp. 159–168.
- Gardner, J. and Belland, B. R. (2012) 'A Conceptual Framework for Organizing Active Learning Experiences in Biology Instruction', *Journal of Science Education and Technology*, 21(4), pp. 465–475. doi: 10.1007/s10956-011-9338-8.
- Gardner, J. L. (2011) *Testing the Efficacy of Merrill's First Principles of Instruction in Improving Student Performance in Introductory Biology Courses*. PhD Thesis. Utah State University. Available at: <https://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=1881&context=etd>.

Garland, R. (1991) 'The Mid-Point on a Rating Scale: Is it Desirable?', *Marketing Bulletin*, 2, pp. 66–70. Available at: http://marketing-bulletin.massey.ac.nz/V2/MB_V2_N3_Garland.pdf (Accessed: 28 June 2021).

Goode, J., Chapman, G. and Margolis, J. (2012) 'Beyond curriculum: the exploring computer science program', *ACM Inroads*, 3(2), pp. 47–53. doi: 10.1145/2189835.2189851.

Google (2018) *Computational Thinking Concepts Guide*, *Google Docs*. Available at: https://docs.google.com/document/d/1Hyb2WKJrjT7TeZ2ATq6gsBhkQjSZwTH-xfpVMFEn2F8/edit?usp=sharing&usp=embed_facebook (Accessed: 20 June 2020).

Google, E. C. T. team (2015) 'Finding Patterns in Spelling Errors and History'. Available at: <https://docs.google.com/document/d/1O6zd-QmVE5CbvCYOpPcWWNnzyONbuqknOU5MuV7KxTg/edit#>. (Accessed: 14 July 2020).

Gough, D., Oliver, S. and Thomas, J. (2012) *An Introduction to Systematic Reviews*. London: Sage.

Grandell, L. (2005) 'High School Students Learning University Level Computer Science on the Web – a Case Study of the DASK-Model.', *Journal of Information Technology Education*, 4, pp. 207–218.

Gravemeijer, K. (1994) 'Educational Development and Developmental Research in Mathematics Education', *Journal for Research in Mathematics Education*, 25(5), pp. 443–471. doi: 10.2307/749485.

Gravemeijer, K. and Cobb, P. (2006) 'Design research from a learning design perspective', in Van den Akker, J. et al. (eds) *Educational Design Research*. Oxford: Routledge, pp. 17–51.

Gravic Inc (2015) 'Career Education Survey', *Remark Software*, 31 July. Available at: <https://remarksoftware.com/sample-forms/2015/07/career-education-survey/> (Accessed: 14 October 2020).

Grover, S. and Pea, R. (2013) 'Computational Thinking in K–12: A Review of the State of the Field.', *Educational Researcher*, 42(1), pp. 38–43. Available at: <https://doi.org/10.3102/0013189X12463051>.

Grover, S., Pea, R. and Cooper, S. (2015) 'Designing for deeper learning in a blended computer science course for middle school students.', *Computer Science Education*, 25(2), pp. 199–237. Available at: <https://doi.org/10.1080/08993408.2015.1033142>.

Guenaga, M. et al. (2017) 'Make world, a collaborative platform to develop computational thinking and STEAM', in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vancouver, BC, Canada, pp. 50–59. doi: 10.1007/978-3-319-58515-4_5.

Guskey, T. R. (2002) 'Does It Make a Difference? Evaluating Professional Development', *Educational Leadership*, 59(6), pp. 45–51.

- Guskey, T. R. (2003) 'What Makes Professional Development Effective?', *The Phi Delta Kappan*, 84(10), pp. 748–750. Available at: <http://www.jstor.org/stable/20440475> (Accessed: 12 September 2018).
- Guskey, T. R. (2016) 'Gauge impact with 5 levels of data.', *Journal of Staff Development*, 37(1), pp. 32–37.
- Guzdial, M. (2008) 'Education Paving the way for computational thinking', *Communications of the ACM*, 51(8), pp. 25–27. doi: 10.1145/1378704.1378713.
- Haberman, B. *et al.* (2011) 'Work in progress—Initiating the Beaver contest on computer science and computer fluency in Israel', in *Frontiers in Education Conference (FIE), 2011*. IEEE, pp. T1D-1.
- Hall, J. N. (2013) 'Pragmatism, Evidence, and Mixed Methods Evaluation', *New Directions for Evaluation*, 2013(138), pp. 15–26. doi: 10.1002/ev.20054.
- Harris, S. (2012) 'Nation can't afford to lose yet another 1,000 jobs in IT sector', *Sunday Independent*, 8 April.
- Harvey, L. (2003) 'Student Feedback [1]', *Quality in Higher Education*, 9(1), pp. 3–20. doi: 10.1080/13538320308164.
- Hidi, S. and Renninger, K. A. (2006) 'The Four-Phase Model of Interest Development', *Educational Psychologist*, 41(2), pp. 111–127. doi: 10.1207/s15326985ep4102_4.
- Holbert, N. R. and Wilensky, U. (2011) 'Racing Games for Exploring Kinematics: A Computational Thinking Approach', in *Proceedings of the 7th International Conference on Games + Learning + Society Conference*. Pittsburgh, PA, USA: ETC Press (GLS'11), pp. 109–118. Available at: <http://dl.acm.org/citation.cfm?id=2206376.2206390> (Accessed: 28 June 2021).
- Holmes, W., Bialik, M. and Fadel, C. (2019) *Artificial Intelligence in Education: Promises and Implications for Teaching and Learning*. Boston: Center for Curriculum Redesign. Available at: <https://curriculumredesign.org/wp-content/uploads/AIED-Book-Excerpt-CCR.pdf>.
- Ioannidou, A. *et al.* (2011) 'Computational Thinking Patterns', in *American Educational Research Association Annual Meeting*, New Orleans, LA: Online Submission. Available at: <https://files.eric.ed.gov/fulltext/ED520742.pdf>.
- ISTE and CSTA (2011b) 'Computational Thinking Teacher Resources Second Edition'. Available at: https://cdn.iste.org/www-root/2020-10/ISTE_CT_Teacher_Resources_2ed.pdf?_ga=2.88148952.278119178.1619789568-1297177301.1619789568.
- ISTE and CSTA (2011a) 'Operational Definition of Computational Thinking for K–12 Education'. Available at: <https://cdn.iste.org/www-root/ct-documents/computational-thinking-operational-definition-flyer.pdf>.
- Izu, C. *et al.* (2017) 'Exploring Bebras Tasks Content and Performance: A Multinational Study.', *Informatics in Education*, 16(1), pp. 39–59.

- Jackson, P. (2001) *Lord of the Rings - Storyboard to Film Comparison - Nazgûl Attack at Bree - YouTube*. YouTube. Available at: <https://www.youtube.com/watch?v=vYjA8xQHK7Q&t=3s> (Accessed: 14 July 2020).
- Jenkins, C. (2015) 'Poem Generator: A Comparative Quantitative Evaluation of a Microworlds-Based Learning Approach for Teaching English', *International Journal of Education and Development using Information and Communication Technology*, 11(2), pp. 153–167.
- Kelleher, C. and Pausch, R. (2005) 'Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers', *ACM Computing Surveys (CSUR)*, 37(2), pp. 83–137.
- Kelly, A. (2004) 'Design Research in Education: Yes, but Is It Methodological?', *The Journal of the Learning Sciences*, 13(1), pp. 115–128. Available at: <http://www.jstor.org/stable/1466935> (Accessed: 23 January 2021).
- Kelly, A. E. (2006) 'Quality criteria for design research.', in Van den Akker, J. et al. (eds) *Educational Design Research*. Oxford: Routledge, pp. 107–118.
- Kelly, A. E. (2013) 'When is Design Research Appropriate?', in Plomp, T. and Van den Akker, J. (eds) *Educational Design Research. Part A: An introduction*, pp. 134–151.
- Kelly, E. O. (2020) *Schools to be given €10m in technology funding*, *RTE News*. Available at: <https://www.rte.ie/news/2020/0422/1134053-education-coronavirus/> (Accessed: 11 June 2021).
- Kennedy-Clark, S. (2013) 'Research by Design: Design-Based Research and the Higher Degree Research student', *Journal of Learning Design*, 6(2), pp. 26–32. doi: 10.5204/jld.v6i2.128.
- KhanAcademy (2014) *Finding the heavier ball*. YouTube. Available at: <https://www.youtube.com/watch?v=0Jgow5x09qw> (Accessed: 16 July 2020).
- Kirwan, C., Costello, E. and Donlon, E. (2018) 'Computational Thinking and Online Learning: A Systematic Literature Review.', in *Proceedings of the: 17th European Conference on eLearning, ECEL 2018*,. Athens, Greece: Academic Conferences and Publishing International Limited, pp. 650–657. Available at: https://www.researchgate.net/publication/329443146_Computational_Thinking_and_Online_Learning_A_Systematic_Literature_Review.
- Kitchenham, B. and Charters, S. (2007) 'Guidelines for performing systematic literature reviews in software engineering', *Engineering*, 2(EBSE 2007-001).
- Koh, K. H. et al. (2014) 'Real time assessment of computational thinking', in *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC*. Melbourne, VIC, Australia, pp. 49–52. Available at: <http://dx.doi.org/10.1109/VLHCC.2014.6883021>.
- Korkmaz, Ö., Çakir, R. and Özden, M. Y. (2017) 'A validity and reliability study of the computational thinking scales (CTS).', *Computers in Human Behavior*, 72, pp. 558–569. Available at: <http://10.0.3.248/j.chb.2017.01.005>

- <http://search.ebscohost.com/login.aspx?direct=true&db=ehh&AN=122721683&site=ehost-live>.
- Krugel, J. and Hubwieser, P. (2017) ‘Computational thinking as springboard for learning object-oriented programming in an interactive MOOC’, in *IEEE Global Engineering Education Conference, EDUCON*. Athens, Greece, pp. 1709–1712. Available at: <http://dx.doi.org/10.1109/EDUCON.2017.7943079>.
- Kuhar, peter (2016) *unrelatedlabs/SpellingCorrector-Java8*, *GitHub*. Available at: <https://github.com/unrelatedlabs/SpellingCorrector-Java8> (Accessed: 14 July 2020).
- Lawrey, S. (2018) *OCR GCSE (9-1) Computer Science - Course Companion*. 2nd edn. Bristol: ZigZag Education.
- Lee, I. *et al.* (2011) ‘Computational thinking for youth in practice’, *Acm Inroads*, 2(1), pp. 32–37.
- Legg, C. and Hookway, C. (2019) ‘Pragmatism’, in Zalta, E. N. (ed.) *The Stanford Encyclopedia of Philosophy*. Spring 2019. Metaphysics Research Lab, Stanford University. Available at: <https://plato.stanford.edu/archives/spr2019/entries/pragmatism/> (Accessed: 13 June 2019).
- Lewis, C., Perry, R. and Murata, A. (2006) ‘How Should Research Contribute to Instructional Improvement? The Case of Lesson Study’, *Educational Researcher*, 35(3), pp. 3–14. doi: 10.3102/0013189X035003003.
- Lexico (2021) *Sort | Definition of Sort by Oxford Dictionary on Lexico.com also meaning of Sort, Lexico Dictionaries | English*. Available at: <https://www.lexico.com/definition/sort> (Accessed: 5 February 2021).
- Liao, L. and Liang, J. (2017) ‘An empirical study on blended learning to promote the development of computational thinking ability of college students’, in *2017 International Symposium on Educational Technology (ISET). Proceedings*. Los Alamitos, CA, USA, pp. 256–60. Available at: <http://dx.doi.org/10.1109/ISET.2017.64>.
- Lifelong Kindergarten Group MIT Media Lab (no date) *About Scratch*. Available at: <https://scratch.mit.edu/> (Accessed: 17 June 2018).
- Lin, P. (2015) *The ethical dilemma of self-driving cars - Patrick Lin*. YouTube. Available at: <https://www.youtube.com/watch?v=ixIoDYVfKA0> (Accessed: 22 July 2020).
- Lincoln, Y. S. and Guba, E. G. (1980) ‘The Distinction between Merit and Worth in Evaluation’, *Educational Evaluation and Policy Analysis*, 2(4), pp. 61–71. doi: 10.2307/1163674.
- Lo, C. K. and Hew, K. F. (2017) ‘Using “First Principles of Instruction” to Design Secondary School Mathematics Flipped Classroom: The Findings of Two Exploratory Studies’, *Journal of Educational Technology & Society; Palmerston North*, 20(1), pp. 222–236.
- Lockwood, J. (2019) *Computer Science To Go (CS2Go): Developing a course to introduce and teach Computer Science and Computational Thinking to secondary school students*. Maynooth University. Available at: <http://mural.maynoothuniversity.ie/11015/>.

Lu, J. J. and Fletcher, G. H. L. (2009) 'Thinking About Computational Thinking', in *Proceedings of the 40th ACM Technical Symposium on Computer Science Education*. New York, NY, USA: ACM (SIGCSE '09), pp. 260–264. doi: 10.1145/1508865.1508959.

Mafumiko, F. (2006) *Micro-scale experimentation as a catalyst for improving the chemistry curriculum in Tanzania*. PhD Thesis. University of Twente. Available at: <https://research.utwente.nl/en/publications/micro-scale-experimentation-as-a-catalyst-for-improving-the-chemi-2> (Accessed: 18 June 2018).

Maguire, M. and Delahunt, B. (2017) 'Doing a Thematic Analysis: A Practical, Step-by-Step Guide for Learning and Teaching Scholars', *All Ireland Journal of Teaching and Learning in Higher Education (AISHE-J)*, 8(3), p. 14.

Maguire, P. and Power, J. (2015) 'Ireland needs to switch on or be left behind in computer science', *The Irish Times*, 20 January. Available at: <https://www.irishtimes.com/news/education/ireland-needs-to-switch-on-or-be-left-behind-in-computer-science-1.2067421> (Accessed: 6 May 2021).

Malan, D. (2014) *Lecture 0 - Introduction to Computer Science I*. Harvard University: YouTube (CS50). Available at: <https://www.youtube.com/watch?v=z-OxZIC6pic&list=PLvJoKWRPIu8G6Si7LlvmBPA5rOJ9BA29R> (Accessed: 16 July 2020).

Marcelino, M. J. *et al.* (2017) 'Learning Computational Thinking and scratch at distance', *Computers in Human Behavior*, 80, pp. 470–477.

Marshall, K. S. (2011) 'Was that CT? Assessing Computational Thinking Patterns through Video-Based Prompts', in *Annual Meeting of the American Educational Research Association (AERA)*. American Educational Research Association, New Orleans, LA.

Mashable Deals (2018) *Google IO developer conference 2018 Google's AI Assistant Can Now Make Real Phone Calls*. YouTube. Available at: https://www.youtube.com/watch?v=JvbHu_bVa_g&feature=youtu.be.n (Accessed: 20 July 2020).

Masole, T. M. (2011) *Enhancing the quality of performance assessment in agriculture in Botswana schools*. Thesis. University of Pretoria. Available at: <https://repository.up.ac.za/handle/2263/28603> (Accessed: 15 June 2018).

Maxcy, S. J. (2003) 'Pragmatic threads in mixed methods research in the social sciences: The search for multiple modes of inquiry and the end of the philosophy of formalism.', in Teddlie, C. and Tashakkori, A. (eds) *Handbook of mixed methods in social and behavioral research*. London, United Kingdom: SAGE, pp. 51–89.

Mayer, R. (2004) 'Should There Be a Three-Strikes Rule Against Pure Discovery Learning?', *The American Psychologist*, 59(1), pp. 14–19. doi: 10.1037/0003-066X.59.1.14.

McKenney, S., Nieveen, N. and Van den Akker, J. (2006) 'Design Research from a Curriculum Perspective', in Van den Akker, J. *et al.* (eds) *Education Design Research*. Oxon: Routledge, pp. 67–90.

McKenney, S. and Reeves, T. C. (2012) *Conducting Educational Design Research*. Abingdon: Routledge.

McKnight, W. and Google Exploring Computational Thinking Team (2015) *Lady Macbeth Chat Bot - Google Docs Whitney McKnight*. Available at: <https://docs.google.com/document/d/1RCP3G2f9QUeRxKGSXcO9QhjX7mK9YDE4Bqeqnh8q3UM/edit> (Accessed: 28 June 2020).

Meerbaum-Salant, O., Armoni, M. and Ben-Ari, M. (2011) 'Habits of programming in scratch', in *Proceedings of the 16th annual joint conference on Innovation and technology in computer science education - ITiCSE '11. the 16th annual joint conference*, Darmstadt, Germany: ACM Press, p. 168. doi: 10.1145/1999747.1999796.

Merriam-Webster (no date) *Definition of Pedagogy*. Available at: <https://www.merriam-webster.com/dictionary/pedagogy> (Accessed: 20 June 2021).

Merrill, D. (2009) 'Finding e³ (effective, efficient, and engaging) Instruction', *Educational Technology*, 49(3), pp. 15–26. Available at: URL: <https://www.jstor.org/stable/44429676>.

Merrill, D. (2013) *First Principles of Instruction. Identifying and Designing Effective, Efficient and Engaging Instruction*. San Francisco: Pfeiffer.

Merrill, M. D. (2002) 'First principles of instruction', *Educational Technology Research and Development*, 50(3), pp. 43–59. doi: 10.1007/BF02505024.

Miles, M. B. and Huberman, A. M. (1994) *Qualitative Data Analysis: An Expanded Sourcebook*. Los Angeles: SAGE Publications Inc.

Miller, L. D. *et al.* (2013) 'Improving learning of computational thinking using creative thinking exercises in CS-1 computer science courses', in *Proceedings - Frontiers in Education Conference, FIE*. Oklahoma City, OK, United states, pp. 1426–1432.

Mingguang, Y. (1999) 'Warm-Up Activities', *English Teaching Forum*, 37(3), p. 24.

Mohaghegh, M. and McCauley, M. (2016) 'Computational Thinking: The Skill Set of the 21st Century', *International Journal of Computer Science and Information Technologies*, 7, pp. 1524–1530.

Moher, D. *et al.* (2009) 'Preferred Reporting Items for Systematic Reviews and Meta-Analyses: The PRISMA Statement', *PLOS Medicine*, 6(7), p. e1000097. doi: 10.1371/journal.pmed.1000097.

Mooney, A. *et al.* (2014) 'PACT: An initiative to introduce computational thinking to second-level education in Ireland', in *International Conference on Engaging Pedagogy 2014. International Conference on Engaging Pedagogy (ICEP) - The Voice of the Educator*, Athlone Institute of Technology. Available at: <http://icep.ie/paper-template/?pid=112>.

Morgan, D. L. (2014) 'Pragmatism as a Paradigm for Social Research', *Qualitative Inquiry*, 20(8), pp. 1045–1053. doi: 10.1177/1077800413513733.

Morreale, P. *et al.* (2012) 'Measuring the Impact of Computational Thinking Workshops on High School Teachers', *J. Comput. Sci. Coll.*, 27(6), pp. 151–157. Available at: <http://dl.acm.org/citation.cfm?id=2184451.2184486> (Accessed: 19 April 2019).

Mouza, C. *et al.* (2017) 'Resetting educational technology coursework for pre-service teachers: A computational thinking approach to the development of technological

pedagogical content knowledge (TPACK).’, *Australasian Journal of Educational Technology*, 33(3), pp. 61–76.

Nagy, W. E. and Scott, J. A. (2000) ‘Vocabulary processes’, in Kamil, M. L., Mosenthal, P. B., and Barr, R. (eds) *Handbook of Reading Research*. Mahwah, NJ: Lawrence Erlbaum Association, pp. 269–284.

National Council for Curriculum and Assessment (NCCA), D. of E. and S. (2016) ‘Short Course Coding Specification for Junior Cycle’. Government of Ireland. Available at: <https://www.curriculumonline.ie/getmedia/cc254b82-1114-496e-bc4a-11f5b14a557f/NCCA-JC-Short-Course-Coding.pdf>.

National Research Council (2010) *Report of a workshop on the scope and nature of computational thinking*. Washington, DC: National Academies Press.

NCCA (2009) *Senior Cycle Key Skills Framework*. Available at: https://www.curriculumonline.ie/getmedia/161b0ee4-706c-4a7a-9f5e-7c95669c629f/KS_Framework.pdf (Accessed: 30 September 2019).

NCCA (2016) *Background Paper and Brief for the development of a new Primary Mathematics Curriculum*. Available at: https://www.ncca.ie/media/1341/maths_background_paper_131016_tc.pdf (Accessed: 30 September 2019).

NCCA (2019) *Primary Developments Final report on the Coding in Primary Schools Initiative*. Available at: https://ncca.ie/media/4155/primary-coding_final-report-on-the-coding-in-primary-schools-initiative.pdf (Accessed: 28 June 2021).

NCCA (no date) *Junior Cycle is changing, Curriculum*. Available at: <https://www.curriculumonline.ie/Junior-cycle/Junior-Cycle-is-changing/> (Accessed: 1 June 2021).

NCCA and DES (2016) ‘Short Course Coding Specification for Junior Cycle’. Government of Ireland. Available at: <https://www.curriculumonline.ie/getmedia/cc254b82-1114-496e-bc4a-11f5b14a557f/NCCA-JC-Short-Course-Coding.pdf> (Accessed: 30 September 2019).

NCCA and DES (2018) ‘Computer Science Curriculum Specification Leaving Certificate Ordinary and Higher Level’. Government of Ireland. Available at: <https://www.curriculumonline.ie/getmedia/d73af6e3-b4e5-4edb-a514-6383e2306a4b/16626-NCCA-Specification-for-Leaving-Certificate-CS-WEB-v4.pdf> (Accessed: 30 September 2019).

NCCA and Primary Developments (2018) *Primary Mathematics Curriculum Draft Specification Junior Infants to Second Class for Consultation*. Available at: https://www.ncca.ie/media/3148/primary_mathspec_en.pdf (Accessed: 30 September 2019).

Nemzer, B. (2011) *Kings, Queens, Jacks & Aces Card Trick | Card Tricks*. YouTube. Available at: <https://www.youtube.com/watch?v=IFOCumySq6g> (Accessed: 15 July 2020).

- Nieveen, N. (1999a) ‘Chapter 10 Prototyping to Reach Product Quality’, in Van den Akker, J. et al. (eds) *Design Approaches and Tools in Education and Training*. Dordrecht: Springer-Science Business Media, p. 125.
- Nieveen, N. (1999b) ‘Prototyping to Reach Product Quality’, in van den Akker, J. et al. (eds) *Design Approaches and Tools in Education and Training*. Dordrecht: Springer Netherlands, pp. 125–135. doi: 10.1007/978-94-011-4255-7_10.
- Nieveen, N. (2010) ‘Formative Evaluation in Educational Design Research’, in *An introduction to educational design research: proceedings of the seminar conducted at the East China Normal University, Shanghai (PR China), November 23-26, 2007*, pp. 89–102.
- Nieveen, N. and Folmer, E. (2013) ‘Formative Evaluation in Educational Design Research’, in Plomp, T. and Nieveen, N. (eds) *Educational Design Research. Part A: An introduction*. SLO • Netherlands institute for curriculum development. Available at: <http://downloads.slo.nl/Documenten/educational-design-research-part-a.pdf>.
- Nieveen, N., Folmer, E. and Vligen, S. (2012) ‘Evaluation matchboard’. SLO • Netherlands institute for curriculum development. Available at: <https://www.slo.nl/publish/pages/4514/evaluatie-matchboard-nl.pdf> (Accessed: 28 June 2021).
- Nieveen, N., McKenney, S. and Van den Akker, J. (2006) ‘Educational design research: the value of variety.’, in Van den Akker, J. et al. (eds) *Educational Design Research*. Oxford: Routledge, pp. 151–158.
- Norvig, P. (2016) *How to Write a Spelling Corrector*. Available at: <http://norvig.com/spell-correct.html> (Accessed: 14 July 2020).
- Othman, M. et al. (2015) ‘Assessing cognitive enhancements in introductory programming through online collaborative learning system’, in *2015 International Symposium on Mathematical Sciences and Computing Research, iSMSC 2015 - Proceedings*. Bandar Meru Raya, Ipoh, Malaysia, pp. 7–12. doi: 10.1109/ISMSC.2015.7594019.
- Oxford Cambridge RSA (OCR) (2020) *GCSE (9–1) Specification Computer Science*. Version 4.3. Available at: <https://www.ocr.org.uk/Images/225975-specification-accredited-gcse-computer-science-j276.pdf>.
- Pane, J. F., Ratanamahatana, C. “Ann” and Myers, B. A. (2001) ‘Studying the language and structure in non-programmers’ solutions to programming problems’, *International Journal of Human-Computer Studies*, 54(2), pp. 237–264. doi: 10.1006/ijhc.2000.0410.
- Papert, S. (1980) *Mindstorms: Children, computers, and powerful ideas*. New York, NY: Basic Books, Inc.
- Patton, M. Q. (2015) *Qualitative Research & Evaluation Methods: Integrating Theory and Practice*. Fourth edition. Thousand Oaks, California: SAGE Publications, Inc.
- PDST (2019) ‘Improving Oral Literacy: Placemat Technique’. Available at: <http://pdst.ie/sites/default/files/Placemat%20Technique.doc> (Accessed: 8 July 2020).
- PDST and DES (2019a) ‘Session 2 More Computational Thinking’. *Leaving Certificate Computer Science National Workshop 4*, September. Available at:

https://www.compsci.ie/fileadmin/user_upload/LCCS_NW4_Session_2_-_Computational_Thinking_reduced.pdf.

PDST and DES (2019b) 'Session 3: Computational Thinking'. *Leaving Certificate Computer Science National WorkShop 3*, January. Available at: https://www.compsci.ie/fileadmin/user_upload/LCCS_NW3_Session_3.pdf.

Pellas, N. and Peroutseas, E. (2016) 'Gaming in Second Life via Scratch4SL.', *Journal of Educational Computing Research*, 54(1), pp. 108–143.

Plomp, T. (2013) 'Design Research: An Introduction', in Plomp, T. and Nieveen, N. (eds) *Educational Design Research. Part A: An introduction*. Enschede: SLO • Netherlands institute for curriculum development, pp. 10–51. Available at: <http://downloads.slo.nl/Documenten/educational-design-research-part-a.pdf>.

Plomp, Tjeerd (2010) 'Educational Design Research: an Introduction', in Plomp, Tj and Nieveen, N. M. (eds) *An introduction to educational design research: proceedings of the seminar conducted at the East China Normal University, Shanghai (PR China), November 23-26, 2007*. 3rd edn. Enschede: SLO • Netherlands institute for curriculum development, pp. 9–36.

Polya, G. (1945) *How to solve it: a new aspect of mathematical method*. 2nd ed. 1990. London: Penguin (Penguin mathematics). Available at: <http://capitadiscovery.co.uk/dcu/items/571788> (Accessed: 7 January 2019).

Preston, V. (2009) 'Questionnaire Survey', in Kitchin, R. and Thrift, N. (eds) *International Encyclopedia of Human Geography*. Oxford: Elsevier, pp. 46–52. doi: 10.1016/B978-008044910-4.00504-6.

Primary Developments and NCCA (2020) 'Draft Primary Curriculum Framework'. NCCA. Available at: <https://ncca.ie/media/4456/ncca-primary-curriculum-framework-2020.pdf>.

Putnam, R. and Borko, H. (2000) 'What Do New Views of Knowledge and Thinking Have to Say About Research on Teacher Learning?', *Educational Researcher*, 29(1), pp. 4–15. doi: 10.3102/0013189X029001004.

Qin Zhang and Baohua Huang (2008) 'How Does Teacher Clarity Affect Student Learning? A Multi-Cultural Test for the Mediated Effect', *Texas Speech Communication Journal*, 33(1), pp. 10–19. Available at: <https://dcu.idm.oclc.org/login?url=http://search.ebscohost.com/login.aspx?direct=true&db=cms&AN=33662917&site=ehost-live&scope=site> (Accessed: 26 August 2021).

Quinlan, A. (2016) 'Teachers and children will need to think in a more computational way in classrooms to be ready for the world of work', *The Irish Examiner*, 1 September. Available at: <https://www.irishexaminer.com/news/arid-20418732.html> (Accessed: 6 May 2021).

Reeves, T. (2006) 'Design Research from a technology perspective', in Van den Akker, J. et al. (eds) *Educational Design Research*. Oxford: Routledge, pp. 52–66.

Repenning, A., Basawapatna, A. and Koh, K. H. (2009) 'Making University Education More Like Middle School Computer Club: Facilitating the Flow of Inspiration', in

Proceedings of the 14th Western Canadian Conference on Computing Education. New York, NY, USA: ACM (WCCCE '09), pp. 9–16. doi: 10.1145/1536274.1536281.

Rodriguez, B. *et al.* (2017) 'Assessing computational thinking in CS unplugged activities', in *Proceedings of the Conference on Integrating Technology into Computer Science Education, ITiCSE*. Seattle, WA, United states, pp. 501–506. Available at: <http://dx.doi.org/10.1145/3017680.3017779>.

Rosenshine, B. and Furst, N. (1971) 'Research on teacher performance criteria', *Research in teacher education*, pp. 37–72.

Rutter, D. *et al.* (2013) 'SCIE systematic research reviews: guidelines', *London: Social Care Institute for Excellence*.

Sabharwal, M., Levine, H. and Dagostino, M. (2016) 'A Conceptual Content Analysis of 75 Years of Diversity Research in Public Administration', *Review of Public Personnel Administration*, 38. doi: 10.1177/0734371X16671368.

Saldana, J. (2016) *The Coding Manual for Qualitative Researchers*. First. London: SAGE. Available at: <https://uk.sagepub.com/en-gb/eur/the-coding-manual-for-qualitative-researchers/book243616> (Accessed: 31 March 2020).

Sandoval, W. (2014) 'Conjecture Mapping: An Approach to Systematic Educational Design Research', *Journal of the Learning Sciences*, 23(1), pp. 18–36. doi: 10.1080/10508406.2013.778204.

Scalable Cooperation and MIT Media Lab (no date) *Moral Machine, Moral Machine*. Available at: <http://moralmachine.mit.edu> (Accessed: 15 June 2021).

Selby, C. C. *et al.* (2013) 'Computational Thinking: The Developing Definition', p. 6. Available at: https://eprints.soton.ac.uk/356481/1/Selby_Woollard_bg_soton_eprints.pdf (Accessed: 28 June 2021).

Selby, C. and Woollard, J. (2014a) 'Computational thinking: the developing definition', *Presented at the SIGCSE 2014, Atlanta*. Available at: <http://eprints.soton.ac.uk/356481/>.

Selby, C. and Woollard, J. (2014b) 'Refining an Understanding of Computational Thinking', p. 23. Available at: <https://eprints.soton.ac.uk/372410/1/372410UnderstdCT.pdf>.

Sener, J. (2015) *Definitions of E-Learning Courses and Programs Version 2.0 April 4, 2015 Developed for Discussion within the Online Learning Community By Frank Mayadas, Gary Miller, and John Sener, Online Learning Consortium*. Available at: <https://onlinelearningconsortium.org/updated-e-learning-definitions-2/>. (Accessed: 28 June 2021).

Sentance, S. and Csizmadia, A. (2017) 'Computing in the curriculum: Challenges and strategies from a teacher's perspective', *Education and Information Technologies*, 22(2), pp. 469–95. Available at: <http://dx.doi.org/10.1007/s10639-016-9482-0>.

Sentance, W. (2014) 'How to write your own spellchecker and autocorrect algorithm in under 80 lines of code', *Medium*, 30 December. Available at: <https://medium.com/@willsentance/how-to-write-your-own-spellchecker-and-autocorrect-algorithm-in-under-80-lines-of-code-6d65d21bb7b6> (Accessed: 9 January 2019).

- Shah, H. (2018) 'Algorithmic accountability', *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 376(2128), p. 20170362. doi: 10.1098/rsta.2017.0362.
- Shavelson, R. J. *et al.* (2003) 'On the Science of Education Design Studies', *Educational Researcher*, 32(1), pp. 25–28.
- Shell, D. F. *et al.* (2014) 'Improving learning of computational thinking using computational creativity exercises in a college CSI computer science course for engineers', in *2014 IEEE Frontiers in Education Conference (FIE). Proceedings*. Piscataway, NJ, USA, pp. 1–7. doi: 10.1109/FIE.2014.7044489.
- Shute, V. J., Sun, C. and Asbell-Clarke, J. (2017) 'Demystifying computational thinking', *Educational Research Review*, 22, pp. 142–158.
- Smidt, S. (2009) *Introducing Vygotsky: A guide for practitioners and students in early years education*. New York: Routledge.
- Smith, H. (2020) 'Algorithmic bias: should students pay the price?', *AI & Soc*, 35(4), pp. 1077–1078. doi: 10.1007/s00146-020-01054-3.
- Smyth, E., McCoy, S. and Banks, J. (2019) *Student, teacher and parent perspectives on senior cycle education*. Research Series Number, 94. ESRI, NCCA. doi: 10.26504/rs94.pdf.
- Stanovich, K. E. (2014) *Rational and Irrational Thought: The Thinking That IQ Tests Miss*, *Scientific American*. doi: 10.1038/scientificamericangenius0115-12.
- Starfield, S. and Ravelli, L. J. (2006) "'The writing of this thesis was a process that I could not explore with the positivistic detachment of the classical sociologist': Self and structure in New Humanities research theses", *Journal of English for Academic Purposes*, 5(3), pp. 222–243. doi: 10.1016/j.jeap.2006.07.004.
- Storte, D. *et al.* (2019) *Coding, Programming and the Changing Curriculum for Computing in Schools*. Report of UNESCO/IFIP TC3 Meeting at OCCE. Meeting at OCCE – Wednesday 27th of June 2018, Linz, Austria: UNESCO/IFIP TC3. Available at: <https://www.ifip-tc3.org/app/download/7193549351/OCCE+2018+TC3+UNESCO+meeting+040219+CS+coding.pdf>.
- Strom, A. R. and Barolo, S. (2011) 'Using the Game of Mastermind to Teach, Practice, and Discuss Scientific Reasoning Skills', *PLoS Biology*. Edited by C. A. Kerfeld, 9(1), p. e1000578. doi: 10.1371/journal.pbio.1000578.
- Sullivan, G. M., Artino, A. R. and Jr (2013) 'Analyzing and Interpreting Data From Likert-Type Scales', *Journal of Graduate Medical Education*, 5(4), p. 541. doi: 10.4300/JGME-5-4-18.
- Sunday Business Post (2012) 'Back to School: Technology is now top of the class', *Business Post*, 16 September. Available at: <https://www.businesspost.ie/legacy/back-to-school-technology-is-now-top-of-the-class-4fbd9ae9> (Accessed: 6 May 2021).
- Sweller, J. (1988) 'Cognitive Load During Problem Solving: Effects on Learning', *Cognitive Science*, 12(2), pp. 257–285. doi: 10.1207/s15516709cog1202_4.

- Sweller, J., Kirschner, P. A. and Clark, R. E. (2007) ‘Why Minimally Guided Teaching Techniques Do Not Work: A Reply to Commentaries’, *Educational Psychologist*, 42(2), pp. 115–121. doi: 10.1080/00461520701263426.
- Taub, R., Armoni, M. and Ben-Ari, M. (2012) ‘CS Unplugged and Middle-School Students’ Views, Attitudes, and Intentions Regarding CS’, *ACM Transactions on Computing Education*, 12(2), pp. 1–29. doi: 10.1145/2160547.2160551.
- TED-ED and Malan, D. (2013) *What’s an algorithm? - David J. Malan*. YouTube. Available at: <https://www.youtube.com/watch?v=6hfOvs8pY1k> (Accessed: 27 July 2020).
- The Design-Based Research Collective (2003) ‘Design-Based Research: An Emerging Paradigm for Educational Inquiry’, *Educational Researcher*, 32(1), pp. 5–8. doi: 10.3102/0013189X032001005.
- The Joint Committee on standards for Educational Evaluation (1994) *The program evaluation standards*. Thousand Oaks, CA: Sage Publications.
- The Open University (2019) *Introduction to computational thinking*. The Open University. Available at: <https://www.open.edu/openlearn/science-maths-technology/computing-ict/introduction-computational-thinking/content-section-0> (Accessed: 28 June 2021).
- The Teaching Council (2020a) *Céim: Standards for Initial Teacher Education, The Teaching Council*. Available at: <https://www.teachingcouncil.ie/website/en/teacher-education/initial-teacher-education/ceim-standards-for-initial-teacher-education/> (Accessed: 30 August 2021).
- The Teaching Council (2020b) ‘Céim: Standards for Initial Teacher Education In accordance with Section 38 of the Teaching Council Acts, 2001-2015’. The Teaching Council. Available at: <https://www.teachingcouncil.ie/en/news-events/latest-news/ceim-standards-for-initial-teacher-education.pdf> (Accessed: 30 August 2021).
- This Morning (2019) *Phillip & Holly Interview This Morning’s First Robot Guest Sophia | This Morning*. Available at: https://www.youtube.com/watch?v=5_jp9CwJhcA (Accessed: 6 April 2021).
- Tianjing, L. and Dickersin, K. (2016) *Introduction to Systematic Review and Meta-Analysis (MOOC)*. Coursera. Available at: <https://www.coursera.org/learn/systematic-review> (Accessed: 15 August 2021).
- Tsai, C.-W. *et al.* (2017) ‘Exploring the effects of web-mediated computational thinking on developing students’ computing skills in a ubiquitous learning environment.’, *Interactive Learning Environments*, 25(6), pp. 762–777.
- Tsai, M.-C. and Tsai, C.-W. (2017) ‘Applying online externally-facilitated regulated learning and computational thinking to improve students learning’, *Universal Access in the Information Society*, pp. 1–10.
- University Of California ActivationLab (2016) ‘Engagement in Science Learning Activities (version 3.2)’. Available at: <http://activationlab.org/wp-content/uploads/2018/03/Engagement-Report-3.2-20160803.pdf>.

University of Pittsburgh (2021) ‘Question Library – University Center for Teaching and Learning’. Available at: <https://teaching.pitt.edu/omet/question-library/> (Accessed: 25 January 2021).

University of Wisconsin-Madison (no date) *Best Practices and Sample Questions for Course Evaluation Surveys, Student Learning Assessment*. Available at: <https://assessment.provost.wisc.edu/best-practices-and-sample-questions-for-course-evaluation-surveys/> (Accessed: 25 January 2021).

Vaismoradi, M., Turunen, H. and Bondas, T. (2013) ‘Content analysis and thematic analysis: Implications for conducting a qualitative descriptive study’, *Nursing & Health Sciences*, 15(3), pp. 398–405. doi: 10.1111/nhs.12048.

Van den Akker, J. *et al.* (eds) (2006) *Educational Design Research*. Oxford: Routledge.

Van der Linden, S., Van der Meij, J. and McKenney, S. (2019) ‘Design and Enactment of Mobile Video Coaching’, *TechTrends*, 63(6), pp. 693–702. doi: 10.1007/s11528-019-00413-2.

Vivian, R., Falkner, K. and Falkner, N. (2014) ‘Addressing the challenges of a new digital technologies curriculum: MOOCs as a scalable solution for teacher professional development.’, *Research in Learning Technology*, 22, pp. 1–19. doi: 10.3402/rlt.v22.24691.

Vygotsky, L. (1978) *Mind in Society: Development of Higher Psychological Processes*. Edited by M. Cole *et al.* Harvard University Press. Available at: https://books.google.ie/books/about/Mind_in_Society.html?id=RxjjUefze_oC&redir_esc=y (Accessed: 29 September 2020).

Waite, J. (2017) *Pedagogy in teaching Computer Science in schools: A Literature Review*. Supplementary Addendum to the Royal Society Computing Education Project Report. Royal Society. Available at: <https://royalsociety.org/~media/policy/projects/computing-education/literature-review-pedagogy-in-teaching.pdf>.

Weintrop, D. *et al.* (2016) ‘Defining Computational Thinking for Mathematics and Science Classrooms’, *Journal of Science Education and Technology*, 25(1), pp. 127–147. doi: 10.1007/s10956-015-9581-5.

Weizenbaum, J. (1966) ‘ELIZA---a computer program for the study of natural language communication between man and machine’, *Communications of the ACM*, 9(1), pp. 36–45. doi: 10.1145/365153.365168.

Weston, C., McAlpine, L. and Bordonaro, T. (1995) ‘A model for understanding formative evaluation in instructional design’, *Educational Technology Research and Development*, 43(3), pp. 29–48. doi: 10.1007/BF02300454.

Wilkerson-Jerde, M. (2014) ‘Construction, categorization, and consensus: student generated computational artifacts as a context for disciplinary reflection.’, *Educational Technology Research & Development*, 62(1), pp. 99–121.

Wing, J. M. (2006) ‘Computational Thinking’, *Communications of the ACM*, 49(3), pp. 33–35.

- Wing, J. M. (2008) 'Computational thinking and thinking about computing.', *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences*, 366(1881), pp. 3717–25. doi: 10.1098/rsta.2008.0118.
- Wing, J. M. (2009) 'Computational Thinking and Thinking About Computing'. *Evening Lecture Series, Florida Institute for Human and Machine Cognition*, Florida, US [Online], 2 September. Available at: <http://www.youtube.com/watch?v=C2Pq4N-iE4I>.
- Wing, J. M. (2011) *Research Notebook: Computational Thinking--What and Why? / Carnegie Mellon School of Computer Science*. Available at: <https://www.cs.cmu.edu/link/research-notebook-computational-thinking-what-and-why> (Accessed: 3 June 2018).
- Winstone, N. E. *et al.* (2017) "It'd be useful, but I wouldn't use it": barriers to university students' feedback seeking and recipience', *Studies in Higher Education*, 42(11), pp. 2026–2041.
- Wood, L. E. (1980) 'An 'intelligent' program to teach logical thinking skills', in *Behav. Res. Methods Instrum. (USA)*. USA, pp. 256–8.
- Xie, B. and Abelson, H. (2016) 'Skill progression in MIT app inventor', in *Proceedings of IEEE Symposium on Visual Languages and Human-Centric Computing, VL/HCC*. Cambridge, United Kingdom, pp. 213–217. doi: 10.1109/VLHCC.2016.7739687.
- Yadav, A. *et al.* (2011) 'Introducing computational thinking in education courses', in *Proceedings of the 42nd ACM technical symposium on Computer science education. SIGCSE '11*, Dallas, Texas, USA: ACM Press, pp. 465–470. doi: 10.1145/1953163.1953297.
- Yadav, A. *et al.* (2014) 'Computational Thinking in Elementary and Secondary Teacher Education', *ACM Transactions on Computing Education*, 14(1), pp. 1–16. doi: 10.1145/2576872.

Appendices

Appendix A: List of Conference Proceedings

- Kirwan, C., (2021) 'Designing a Computational Thinking Course for Irish Second Level Students and Teachers', *DCU Postgraduate Research Unconference*. Online. June 3rd.
- Kirwan, C., (2021) 'Teaching and Learning Computational Thinking in the Irish Classroom', *CESI Conference 2021: Adapting Digitally to a Changing Reality*. Online. February 2021.
- Kirwan, C., (2020, May) 'Design Based Research: The Highs, Lows and Compromises', *DCU Postgraduate Research Unconference*. Online. 27th May.
- Kirwan, C., (2020,) 'Design Based Research and Computational Thinking: The Highs and Lows', *Trinity College PG Student Research Conference*. Online. 23rd May.
- Kirwan, C., (2019) 'Design-Based Research and Computational Thinking. An introductory Computational Thinking course for post-primary students', *Post-Graduate Research Conference (School of Education)*. Trinity College. Dublin, Ireland. 17th May.
- Kirwan, C., (2020) 'Computational Thinking Unplugged', *CESI Conference 2020 Our Evolving Learning Landscape*. Athlone, Ireland. 29th February.
- Kirwan, C., (2019) 'Designing an introductory Computational Thinking Course for post-primary students', *Educational Studies Association of Ireland (ESAI)*. Sligo, Ireland. 13th April.
- Kirwan, C., (2018) 'The machine in the ghost: Teaching computational thinking to Irish students. Paper in student session on Education & Learning', *1st Irish Postgraduate Research Conference (IPRC)*. Dublin, 8th November.
- Kirwan, C., (2018). 'The machine in the ghost: Teaching computational thinking to second level students using a Massive Open Online Course (MOOC). Finalist at DCU Tell it Straight Competition, available at <https://www.youtube.com/watch?v=FzO2xNGoGAc&t=17s> Dublin, 7th March.
- Kirwan, C., (2017) 'Computational thinking for Post Primary students', Presentation at *Computers in Education Society of Ireland (CESI) Conference*. St Patrick's Campus, Dublin City University, Dublin, 4th March.
- Kirwan, C., Costello, E., & Donlon, E. (2017) 'Complexities of a computational thinking age: Definitional challenges', Presentation at *EdTech 2017 TEL in an Age of Supercomplexity - Challenges, Opportunities and Strategies*. Annual Conference of the Irish Learning Technology Association, Sligo, 1st & 2nd June.
- Kirwan, C., Costello, E., & Donlon, E. (2017) 'Computational thinking: Are we all on the same page?', Presentation at *Education Studies Association Ireland (ESAI) Conference*. University College Cork, Cork, 21st April.
- Kirwan, C. (2016) 'Teaching computational thinking to Irish post-primary school students: A doctoral study in progress', Concise paper at *The Next Generation Digital Learning Research Symposium*. Dublin City University, 1st November.

Appendix B: Engagement Questionnaire

This was adapted from the University of California's Activation lab Engagement Survey (2016).

Student Post Lesson Questionnaire

Please indicate your level of agreement with the following statements.

1. During this lesson: I felt bored.
Strongly Disagree Disagree Neutral Agree Strongly Agree
2. During this lesson: I felt happy.
Strongly Disagree Disagree Neutral Agree Strongly Agree
3. During this lesson: I felt excited.
Strongly Disagree Disagree Neutral Agree Strongly Agree
4. During this lesson: I was daydreaming a lot
Strongly Disagree Disagree Neutral Agree Strongly Agree
5. During this lesson: I was focused on the things we were learning most of the time.
Strongly Disagree Disagree Neutral Agree Strongly Agree
6. During this lesson: Time went by quickly.
Strongly Disagree Disagree Neutral Agree Strongly Agree
7. During this lesson: I was busy doing other tasks.
Strongly Disagree Disagree Neutral Agree Strongly Agree
8. During this lesson: I talked to others about stuff not related to what we were learning.
Strongly Disagree Disagree Neutral Agree Strongly Agree

Gender: Please circle the appropriate option.

Male Female

Appendix C: End of Course Questionnaire

Questionnaire issued at end of V3:

During the Computational Thinking course:

Activation (Unplugged activating previous learning)

I engaged in activities that helped me learn ideas or skills that were new and unfamiliar to me.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

In this course, I was able to connect the activities to new ideas and skills I was learning.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

Demonstration

Media used in this course (for e.g. videos, websites,slides) were helpful in learning.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

My teacher gave examples of concepts that I was expected to learn.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

Unplugged Evaluation

The activities used in this course were helpful in learning.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

The activities helped increased my knowledge and skills in Computational Thinking.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

Application

I had opportunities to practice what I learned in this course.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

In this course, I was able to get feedback on my work

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

I had the opportunity to test or evaluate my solutions to activities

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

Real-World

I had the opportunity to work on problems that occur in real-life.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

Integration

I see how I can apply what I learned in this course to other subjects

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

I see how I can apply what I learned in this course to Computer Science

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

In this course, I was able to show my teacher my work

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

I had opportunities to reflect or discuss what I learned in this course.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

Satisfaction

Compared to what I knew before I took this course, I learned a lot.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

I did not learn a lot in this course.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

I am very satisfied with the content of this course

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

I am dissatisfied with the activities used in this course

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

This course was a waste of time.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

Quality

Overall, I would rate the quality of this course as outstanding.

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

Overall, I would recommend this course to other students

Strongly Disagree *Disagree* *Neutral* *Agree* *Strongly Agree*

Q1 What do you think you learned during the course?

Q2 What are the strengths of this course?

Q3 Do you have any specific recommendations for improving this course?

Generic Questions: Please circle the correct answer

Q1. I have previous experience of a computer programming language

Yes No

If Yes please state the name of the programming language

Q2. I have previous experience of Computer Science for example Robotics, web development, did classes in school

Yes No

Please give details

Appendix D: Pre and Post Tests (Version 1)

Q1a What do you think Computational Thinking might mean?

Q1b Can you describe any situations/problems that might use Computational Thinking?

Q2a Describe what you understand by Artificial Intelligence?

Q2b What computer programmes do you think may use Artificial Intelligence or are Artificial Intelligence programmes?

Q3a Explain what you understand by algorithms?

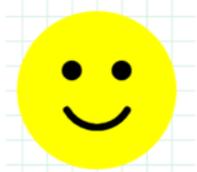
Q3b Give or guess some examples of algorithms.

Q4a: You are a passenger on a car journey. The driver is concerned that (s)he may have taken a wrong turn. The road you are on is more like a boreen (narrow country road, with no road markings), (s)he was expecting the road to be the main road. You and the driver have not travelled this journey before. The Sat Nav tells you, that you are going in the right direction and that you are on the right road.

What would you advise the driver to do?

Q5 Imagine that you are playing a drawing game. You are paired with a partner. You are given the following picture. Your partner does not see this picture.

Output



The objective of the game, is that your partner must draw the above picture of the smiley face by following instructions from you. You are not allowed to tell them what it is. Can you write down the instructions you would give to replicate this picture?

Post Test:

Number: _____

Q1a What is Computational Thinking?

Q1b Describe in simple terms how would you use some (or all) the components of Computational Thinking to help solve the following problem, -- Creating a study timetable

Q2a Describe what you understand by Artificial Intelligence?

Q2b What computer programmes, or types of computer programmes do you think may use Artificial Intelligence?

Q3a Explain what you understand by algorithms?

Q3b Give or guess some examples of algorithms.

Q4a:

Do ethics have a place in Computational Thinking? Please provide an example to explain your reasoning.

Q 4b Does Logic have a place in Computational Thinking? Please give an example to explain your reasoning.

Q5a Problem

Imagine that you are playing a drawing game. You are paired with a partner. You are given the following picture. Your partner does not see this picture.



The objective of the game, is that your partner must draw the above picture of a house by following instructions from you. You are not allowed to tell them what it is. Can you write down the instructions you would give to replicate this picture?

Generic Questions: Please circle the correct answer

Q7 a I have previous experience of a computer programming language

Yes No

Q7b If Yes please state the name of the programming language

Q8 I have previous experience of Computer applications such as.

Please circle the correct options

MS word: Yes No

Email: Yes No

Spotify: Yes No

Netflix: Yes No

Q9 I have previous experience of Computer Science for example Robotics, web development

Yes No

Please give details

Appendix E: MCQ Tests

8. Computational Thinking is best defined as 1 point

Mark only one oval.

- Thinking like a computer
- Learning how to program using logic and patterns
- Making a computer use artificial intelligence
- Using skills from Computer Science to help solve problems

9. The breaking down of a problem into smaller problems is an example of 1 point

Mark only one oval.

- abstraction
- decomposition
- pattern matching
- algorithm creation

10. Which of the following is an example of thinking computationally? 1 point

Mark only one oval.

- Experimenting with different materials to see which is the best type for your garment (an item of clothing)
- Planning how to sew the garment and devising a list of steps to follow
- Asking an expert how to sew a garment
- Getting an expert to create and sew the garment for you

11. Check all options that apply. Computers are very good at

1 point

Tick all that apply.

- Performing simple instructions very fast
- Thinking of different ways to solve problems
- Running programs that were written by humans

12. This passage is from Dr John Snow, who traced the source of a cholera outbreak "On proceeding to the spot, I found that nearly all the deaths had taken place within a short distance of the [Broad Street] pump. There were only ten deaths in houses situated decidedly nearer to another street-pump. In five of these cases the families of the deceased persons informed me that they always sent to the pump in Broad Street, as they preferred the water to that of the pumps which were nearer" The above passage illustrates a particular skill of computational thinking. Which one is it?

1 point

Mark only one oval.

- Pattern Matching
- Decomposition
- Algorithm design
- Abstraction

13. Capturing the essential points, and removing unnecessary details is an example of what?

1 point

Mark only one oval.

- Decomposition
- Abstraction
- Pattern Matching

14. Identify which statement describes algorithmic thinking 1 point

Mark only one oval.

- Identifying patterns and documenting their essential characteristics
- Identifying the steps involved in solving a problem
- Identifying what problems need to be solved
- Identifying how to convert a letter to a binary number

15. Check all that apply. The following are examples of algorithms 1 point

Tick all that apply.

- A Cake Recipe
- A Computer program
- Lego instructions
- Computer memory

16. Check all the apply. Algorithms consist of 1 point

Tick all that apply.

- unambiguous (clear) instructions
- instructions written in order of execution
- instructions written only in a computer language

17. The process of arriving at a valid decision based on the evidence in front of you is called 1 point

Mark only one oval.

- logical thinking
- computational thinking
- algorithmic thinking

18. Check all the true statements. Logical Thinking is an important part of Computational Thinking as 1 point

Tick all that apply.

- it helps to ensure the validity of algorithms
- computers can think
- computers are intelligent

19. Searching a library, one book at a time, is an example of 1 point

Mark only one oval.

- Linear Search
- Computational Search
- Binary Search

20. The data for binary search 1 point

Mark only one oval.

- must be letters
- must be numbers
- must be sorted in a list
- must be unsorted in a list

21. A list of number contains the following seven values 0, 2, 3, 6, 7, 9, 15. How many values would a binary search examine before it found the value 6? 1 point

Mark only one oval.

- 7
- 4
- 1
- 2

22. Select the best answer. Artificial intelligence is concerned with 1 point

Mark only one oval.

- Making a computer intelligent by increasing its memory
- false intelligence
- machines that learn by feedback and pattern matching
- machines that are self reliant i.e. need no human intervention and are self aware

23. The term Ethics is concerned with 1 point

Mark only one oval.

- Making decisions that are valid
- Making decisions that are moral i.e. good and bad
- Making decisions that generate money

24. Check all the true statements. Ethical Thinking is 1 point

Tick all that apply.

- An important part of Computational Thinking because computer algorithms are involved in decisions that can affect our lives
- Essential to Computational Thinking as computer algorithms are written by humans
- Only essential in Medicine
- Redundant in Computational Thinking as computer algorithms are run on machines

25. The Turing Test was devised to establish if 1 point

Mark only one oval.

- A computer runs at optimum (i.e. best) performance
- A human is smarter than a computer
- A computer can understand language

26. Check all that are true. Pseudo-code is a language

1 point

Tick all that apply.

- that can run on all computers
- is intended for human reading only
- uses the structure of a programming language

27. The following algorithm should take as input two numbers. It should then add together these two number and output the result. Identify the correct algorithm. 1 point

A)
`num1 = input("Enter the first number")
num2 = input("Enter the second number")
num3 = num1 + num2
print(num3)`

B)
`num1 = input("Enter the first number")
num3 = input("Enter the second number")
num3 = num1 + num2
print(num3)`

C)
`num1 = input("Enter the first number")
num2 = input("Enter the second number")
num3 = num1 + num2
print(num2)`

Mark only one oval.

- A
- B
- C

28. The following algorithm uses a decision i.e. a conditional statement. Identify the words it outputs to the screen.

1 point

```
answer = 10;
if ( answer < 20)
    print ("Under");
else
    print ("Over");
```

Mark only one oval.

- Over
- Under
- Under Over

Appendix F: Course Glossary

PlaceMats

The following is a snippet from the Teacher guide that explains how placemats are used

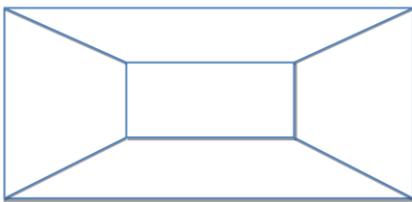
Activity One.

Learning Goal: Introduce students to the concepts of decomposition, abstraction and pattern matching

Divide students into groups of four/five. Hand each group a bag of buttons (which contains buttons of different shapes and sizes) and an A3 placemat or similar (see

<http://pdst.ie/sites/default/files/Placemat%20Technique.doc>).

The placemat was recommended by a teacher in prototype one and proved successful for capturing students' ideas. Amend the template to suit the group size. The below pictures shows a template for a group of four.



Ask students to:

- individually consider how they would group the buttons, (based on the characteristics of the buttons that they think are important).
- write their idea in their individual placemat section
- discuss their idea with the group and come up with a consensus categorisation plan.
- categorise the buttons and write their final idea in the placemat when they are finished (as they may change their mind during the process.)

This activity will serve to show students that they automatically used a strategy that consisted of :

- decomposing the buttons, i.e. putting them into groups. These groups were based on shapes, colours, i.e. they performed pattern matching. Their groups were formed based on essential details, i.e. abstraction

Reflection

Ask each group to state what they did and why. Highlight to students how they used

pattern matching, to break down the problem and to identify the essential characteristics. They used a strategy that involved skills from computer science: decomposition, pattern matching and abstraction. Just think about how you make a jig-saw puzzle, you group the pieces and then assemble them. How you group the pieces is different for each person. Some people group all the edges together first, others look for the corners and work from there, others look for a specific image in the jig-saw and identify all the pieces associated with that component. What is significant here, is that you created groups based on patterns that you identified were important to get the job done.

(Extra information: this task may also serve to explain to students the difference between sorting (arranging in order) categorising (grouping) and also tagging (the categorisation may have sub-groups).

Thirty Second Game:

The following is a snippet from the Teacher guide that explains the thirty second game

Activity Two. A game of thirty seconds.

In their groups, each student will get a chance to play this game.

The game is played as follows. Each thirty second card contains five words. In their groups, each student, in turn, has thirty seconds to describe the five words on their card to their group. The group must guess the word.

When describing the 'word', the student cannot say the word on the card, or a part of the word (this includes a translation of the word), or use pointing, when giving their descriptions. For example, if the word is 'chips': you might say Fish and ????. Or fried potatoes are called ???.

The teacher first demos the game for the students and then will time each game. When all students have played, they write (in their placemat) an example of the best description that was used to guess a word, and what it was based on, for example, opposites, characteristics etc.

Appendix G: Summary 'On The Fly' Revisions

Topic	Changes made to Unit 1	Criterion addressed
Unit 1	Two new pre-activation puzzles added, button sorting and the 30-second activity. "Placemats" added to aid collaboration, and to involve students more in the reflection of activities The Logic puzzles moved to the start of the course	Student Engagement Student Understanding
	The number of slides reduced from 31 to 14. Removed examples of CT from everyday life as too confusing Screencast of spelling activity code created	Practicality of materials
	Spelling activity, scaffolded more, as cognitive load too high	Unplugged activities Student Engagement

Topic	Changes made to Unit 2	Criterion addressed
Unit 2	Backup created of all videos (in case youtube not available) Create screencasts of social media tweets, as Twitter not allowed in some schools Less detail in slides, bigger writing	Practicality
	Changed the codemaker worksheet for Colorado School of Mines to add clarity	Engagement
	Add recap (to the previous lesson) at the start of all lessons Refine Logic puzzle to improve clarity, for example, with the logic exercise, let students know that they can refill the bottles with more water during the task. Highlight more the linkage of activities (logic and algorithms) to Computational Thinking	Unplugged Activity

Topic	Changes made to Unit 3	Criterion addressed
Unit 3	Develop templates for Searching to Speak activity as too abstract, lacks clarity. Add in a video before "Searching to Speak" demo, as a lead-in/ hook, to activity Create a fact sheet for teachers about Dominique, as students wanted more information	Unplugged Activities
	Engagement puzzles at the start of the lesson, now have a dual function to serve as a pre-activation task	Engagement Understanding

	Highlight the relevancy of the weighing ball engagement puzzle to both Computational Thinking and Computer Science	Student Understanding
--	--	-----------------------

Topic	Changes Made to Unit 4	Criterion addressed
Unit 4	Need to make chatbot exercise clearer. Provide sample answers	Unplugged Activity

Topic	Changes Made to Unit 4	Criterion addressed
Lesson 5	Update Cheatsheets, to provide more code snippets and examples Update Cheat Sheets, so options similar to that provided in Scratch	Unplugged Activities
	Remove exercise to write instructions to make cup of tea. Lack of engagement	Student Engagement

Appendix H: Engagement scores Version 1

School 1

Week 1 N=16	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	3	4	4	3.5	4	5
Mode	4	4	3	4	4	3	4	5
Week 2 N=9	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	5	4	4	4	4	5	4	4
Mode	5	4	4	4	4	5	5	5
Week 3 N=16	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4.5	4	4	4	4	4	4	4
Mode	5	4	4	4	4	4	4	4
Week 4 N=13	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	3	4	4	4	4	4
Mode	4	4	3	4	4	4	4	2*
Week 5 N=9	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	4	4	4	4	4	5
Mode	5	4	4	4	4	4	4	5

® = reversed, * = multimode, smallest value shown

School 2

Week 1 N=13	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	4	4	4	4	4	3
Mode	4	4	4	4	4	4	4	3
Week 1 N=15 2 nd Cycle	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	4	4	4	4	4	3
Mode	4	4	4	4	4	4	3	2*
Week 2 N=25	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	4	4	4	4	4	3
Mode	4	4	4	4	4	4	4	4
Week 3 N=28	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®

Median	4	4	4	4	4	4	4	4
Mode	4	4	4	4	4	4	4*	5
Week 4 N=22	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	4	4	4	4	4	4
Mode	4	4	4	4	4	4	4	4

® = reversed, * = multimode, smallest value shown

School 3

Week 1 N=28	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	4	4	4	4	4	4
Mode	4	4	4	4	4	4	4*	5
Week 2 N=22	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	3	4	4	4	4.5	4
Mode	4	4	3	4	4	4	5	5
Week 3 N=28	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	4	4	4	4	4	4
Mode	4	4	4	4	4*	4	5	5
Week 4 N=24	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	5	4	4	4	4	4	4	4
Mode	5	4	4	4	4	4	5	5
Week 5 N=24	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	3	4	4	4	4	3
Mode	4	4	3	4	4	4	4	2*

® = reversed, * = multimode, smallest value shown

Week 1 N=5	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	3	4	4	4	4	4
Mode	4	4	3	4	4	4	4	4
Week 2 N=5	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	3	4	4	4	4	4
Mode	4	4	3	4	4	4	4	4
Week 3 N=8	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®

Median	4	4	3.5	3.5	4	4	4	3
Mode	4	4	4	4	4	4	4	3*
Week 4 N=9	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	4	4	4	4	4	4	4	3
Mode	4	4	4	4	4		4	4
Week 5 N=8	Bored ®	Happy	Excited	Daydream ®	Focused	Time Quick	Busy Other ®	Talk ®
Median	3	3.5	3.5	3	4	3	3	2
Mode	4	3*	2*	3	4	3	3	2

® = reversed, * = multimode, smallest value shown

Appendix I: Revisions from Validation Stage

Category: Sub Category:	Corrective Feedback List	Action Made
Quality: Interview Questions	Consider the word relevant in your interview question, relevant to what? Does the content of the curriculum include relevant subjects and topics?	I changed my interview script to accommodate this feedback.
Quality: Interview	Ensure, I cover my review of content from multiple angles	I had five experts and eleven teachers read my course
Corrective Feedback Content Language:	Change the course document's title to give context, such as course length and the type of students it is aimed at.	I changed the title from: An introductory course to Computational Thinking to A short introductory course in Computational Thinking for Transition Year students and Teachers.
Corrective Feedback: Content Language	Problem: Giving multi-examples to explain a concept “pick up on a word or a term that was in the computational thinking orbit shall we call it and then you would elaborate every meaning of it.”	Need more computational examples. From discussions with teachers, it was clear that Computer Science is the baseline. Hence I gave more Computer Science examples of Computational Thinking components I also reduced the complexity of my slides by providing fewer examples. This was something that I did organically in the course
Corrective Feedback: Content Language	Computer Science meaning Vs real-world meaning of the word “sorting”. In computer terms, this means sort in order, in non-computer terms it means categorisation.	I updated the document to use the word categorise, and I explained the difference of the word sorting in computer and non-computer terms in same document
Corrective Feedback: Content	AI and ethics: “Is it relevant in a tightly packed course?”	I specified my reasoning for including this topic in the course document. It is related to Leaving Cert (computer Science in Society) and . Computational Thinking needs to be both valid and morally correct.
Corrective Feedback: Content	Data: “Include a topic on Data Representation?” “The structuring of data, the choosing of data, the way data is stored, the way data is dealt with, the way data is designed actually is”	I made my data examples more explicit and highlighted them in the programming/pseudocode, Computational Thinking and the AI and ethics part.
Corrective Feedback: Content	Computational Thinking FlowChart “Does Computational Thinking occur in a Linear Fashion?”	Reflecting on this feedback using experience from the first piloting of the course, the flow chart in week 1 was reconfigured to remove the sequential order of Computational Thinking elements. They are now listed in a way that does not imply order. (see
Corrective Feedback: Content	Need a Curriculum Map for overview of contents	The document now has a curriculum map

Corrective Feedback: Content-unplugged activities	More exemplification of CT concepts. CT Theoretical ideas are baffling. More relevant examples needed	More real-world computer science examples were introduced for example, code examples of ChatBot. The spelling example was expanded and made more scaffolded to highlight the CT concepts. Jigsaw puzzles were introduced to explain decomposition.
Corrective Feedback: Content Subjects	In the algorithm section at week 2 I did not break down algorithmic design to include repetition, input and sequence ,	No Action taken to content in week 2, as this is done in the programming section, and an explanation added for teachers to explain it is dealt later on
Corrective Feedback: Content Subjects	Explain to students : “How is it that it’s relevant to know that a computer blindly follows instructions? “	No Action taken; This is tackled in week 2, week 4 and week5. I explain that computers are not intelligent they are just an extension of our reasoning
Corrective Feedback: Content Subjects Language	Subject and Language Algorithm characteristics “Finite, self-working ?? as students may know Scratch, so we say programming are finite:- there is a forever operation. Do we need to think about parallel processing?”	No. I kept it simple and mentioned characteristics like finite and self-working as this is an introductory course. I did change my text to say qualities rather than essential characteristics
Corrective Feedback: Content: Language	Have definition of CT high in the document and give to students and teachers. Justify definition	This was implemented
Corrective Feedback: Content: Subjects	AI: what am I doing? AI in society or ethics of AI etc How are CT and AI related?	I rewrote this topic to explicitly express the relationship between AI and CT.
Corrective Feedback: Content: Subject	AI: “Ensure there are no misconceptions” “Ensure they understand that AI is still deterministic”	I rewrote the AI topic to ensure that I expressed that AI programmes still follows “rules”, that the computer is not intelligent I added in a definition for AI. I had explained it in general but did not refer to a definition

Appendix J: Pre and Post Test Rubrics (Version 1)

CT Definition in course.

So what is Computational Thinking? Well, simply put it is a framework for problem-solving, that uses concepts from Computer Science. In Computer Science, when analysing and solving problems, one has to think a certain way. This reasoning is very much influenced by the way computers can be programmed, (but as stated earlier, this reasoning can be applied to other disciplines, such as science, mathematics and humanities).

These conceptual skills are collectively known as Computational Thinking. Four of the core skills of Computational Thinking are decomposition, pattern matching, abstraction and algorithms.

Logic will also be presented. Logical thinking helps to ensure the validity of our reasoning. The overall goal of this lesson is to highlight to students that computers do not think for themselves; they are an automation of our reasoning, and that reasoning should be valid.

CT Unsatisfactory	CT Partial proficient	CT Proficient
<p>The student's answer of CT did not subscribe in any part to what was taught in the course.</p>	<p>Student's answer is partially correct in that their description of CT matches partly what was taught in that course but did not fully subscribe to the definition.</p>	<p>Student's answer subscribes to what was taught in the course. It recognises that CT is a thought process. That there are components of CT: decomposition, pattern matching, abstraction and algorithms. Logical thinking is also a part of CT as it helps to ensure the validity of our reasoning.</p> <p>Answers may also allude to the following <i>In Computer Science, when analysing and solving problems, one has to think a certain way. This reasoning is very much influenced by the way computers can be programmed</i></p>

Algorithms Rubric:

Algorithms		
<p>Characteristics and qualities of an algorithm described in the course Algorithms are a series of steps that are precisely defined.</p> <ul style="list-style-type: none"> • They are consistent, i.e. based on the input, you should know what the output should be. • These steps are sequential • The steps can be considered “an instance in time,” (Beecher, 2017, pp. 26–27) i.e. once a step is executed, it is forgotten about. • The user does not need to understand the steps. (Although in the ethics section we will discuss the need for the transparency of algorithms!!!). 		
Algorithms Unsatisfactory	Algorithms Partially proficient	Algorithms Proficient
<p>The student’s answer of algorithms does not subscribe in any part to what was taught in the course.</p>	<p>The student’s answer is partially correct in that their description of algorithms matches partly what was taught in that course but may contain some inaccurate information</p>	<p>The student’s answer subscribes to what was taught in the course.</p> <p>It recognises that algorithms are instructions that are defined as a series of steps, that have a purpose, for example solve a problem</p> <p>Other points to consider is that student knows some of the characteristics of good algorithms They are consistent. These steps are sequential The steps can be considered “an instance in time,” i.e. once a step is executed, it is forgotten about (Beecher, 2017, pp. 26–27).</p>

Artificial Intelligence Rubric

OLD Definition

‘Artificial intelligence (AI) refers to systems that display intelligent behaviour by analysing their environment and taking actions – with some degree of autonomy – to achieve specific goals’ (European Commission High level Expert Group on Artificial Intelligence, 2019, p. 1)

New Definition

‘Artificial intelligence is an entity (or collective set of cooperative entities), able to receive inputs from the environment, interpret and learn from such inputs, and exhibit related and flexible behaviors and actions that help the entity achieve a particular goal or objective over a period of time’ (Faggella, 2018).

The AI machine does not think for itself, but it does use machine learning algorithms to **learn the best patterns to use**, i.e. the patterns were not pre-programmed. But of course these machine learning algorithms were written by humans. It is the humans that are intelligent; the computer is following our rules.

“You give the machine data, a goal and feedback when it’s on the right track – and leave it to work out the best way of achieving the end” (Hannah Fry (2018) pg 10-11).

AI Unsatisfactory	AI Partial Proficient	AI Proficient
<p>The student’s answer of AI does not subscribe in any part to what was taught in the course.</p>	<p>Student’s answer is partially correct in that their description of AI matches partly what was taught in that course but does not fully subscribe to the definition.</p> <p>Uses data, receives input, achieve a goal</p>	<p>Student’s answer subscribes to what was taught in the course.</p> <p>It recognises that AI systems display intelligent behaviour but are not “thinking” bodies. It recognises that AI agents receive inputs for example sensor or data and makes decisions based on that data. These agents receive feedback based on their decisions</p>

Appendix K: Logic and Ethics Results (Version 1)

School	Question	Post Results	Comments
1	Logic	11Y, 1N	Seven of the 'Yes' answers referred to validity. Student 15 states that 'Logic was the component that made everything start to make sense' but no rationale given. The one 'No' answer is as follows: 'No, as logic can be used more in real life'. This answer suggest that the student was not cognizant of the role logical thinking plays in the validity of algorithms
2	Logic	19Y, 2N	Eleven of the Yes answers provided validity as a rationale for logical thinking in connection with CT. Two students, in their answers made mention of efficiency and computational power . I suspect they were making reference to the computer using its computational power and human logic to solve the Codebreaker puzzle. Whilst their answer is correct, the connection to validity would be considered more correct under the lens of this course
3	Logic	22Y, 2N	<p>Only two responses disagreed with logic having a place in Computational Thinking. Both answers showed the students were confused: 'no, because computers do what they are programmed do and don't take the place of thought' (Student S40). The other student had missed three units, and did not understand what logic meant.</p> <p>Regarding the 'Yes' answers, Nine answers presented vague rationale, whilst ten answers related logic with common sense and validity of algorithms</p>
1	Ethics	9Y,3N	<p>Six of the 'Yes' responses provided rationale. Four alluded to the driverless car: 'Ethics are important in computational thinking as whenever an advancement is made in computing we must question whether or not it is ethical for example the safety precautions in a driverless car' (Student A1).</p> <p>There was some evidence of misunderstanding: 'Not really because a computer can't decide for itself what's right or wrong (Student A5)'</p>

2	Ethics	20Y	Eleven of the answers specifically mentioned the driverless car: 'Yes they do as long as computers in a position in which they can affect human lives in negative way for example self driving cars (student A123). 'Ethics do have a place in computational thinking. Ethics could be involved in the programming a driverless car (A120)'
3	Ethics	12Y, 9N	<p>Twelve 'Yes' answers. Nine 'No' answers. Of the students that said 'Yes', six related answers to the driverless car, two with privacy, and three to programmes having to be moral.</p> <p>With reference to the 'No' answers: four did not know what ethics was with one student confusing ethics with work ethic.</p>

Appendix L: Original Assessment Plans

Unit 1:

Number	Learning Outcomes
LOT1-1	Critically assess problem-solving using both a strategy and guesswork.
LOT1-2	Identify the components of the Computational Thinking framework
LOT1-3	Apply elements of the Computational Thinking framework to help solve a problem.

Number	Name	Description	Assessment Type
1A-1	Week 1 MCQ1	MCQ CT Description	GDrive Form Topic1_PreTest. MCQ
1A-2	Week 1 MCQ2	Decomposition Q	MCQ
1A-3	Week 1 MCQ3	CT Description	MCQ
1A-4	Week 1 MCQ4	Computing understand	MCQ
1A-5	Week 1 MCQ5	Pattern Matching	MCQ
1A-6	Week 1 MCQ6	Abstraction	MCQ
1A-7	Marry Puzzle	Puzzle highlight cognitive laziness	attitude change Teacher Observation
1A-8	Bat Puzzle	Puzzle highlight cognitive laziness	attitude change Teacher Observation
1A-9	Button Activity	Students individually write how they would sort the button. As a group they decide on final categorisation	Portfolio: Placemat
1A-10	Thirty Seconds	Students' experiences abstracting a problem. Students identify a good abstraction example	Portfolio: Placemat
1A-11	Spell Checker	Decompose problem, identify steps. Notice patterns in the spellings	Portfolio: Placemat, Teacher Observation, did students notice the patterns etc.
1A-12	Week 1 MCQ1	MCQ CT Description	GDrive Form Topic1_POST_Test. MCQ
1A-13	Week 1 MCQ2	Decomposition Q	MCQ
1A-14	Week 1 MCQ3	CT Description	MCQ
1A-15	Week 1 MCQ4	Computing understand	MCQ
1A-16	Week 1 MCQ5	Pattern Matching	MCQ
1A-17	Week 1 MCQ6	Abstraction	MCQ

Activity	Learning Outcome	Bloom (B) /Skill/Affective	Description
1A-1	L0T1-2	B Comprehension	
1A-2	L0T1-2	B Comprehension	
1A-3	L0T1-2	B Application	
1A-4	L0T1-2	B Knowledge	
1A-5	L0T1-2	B Analysis	
1A-6	L0T1-2	B Comprehension and Knowledge	
1A-7	L0T1-1	AFFECTIVE LEARNING	Puzzle: Learning activity to highlight strategy better than guesswork: Teacher confirms if this achieved, asks students after puzzles
1A-8	L0T1-1	AFFECTIVE LEARNING	Puzzle: Learning activity to highlight strategy better than guesswork: Teacher confirms if this is achieved, asks students after puzzles
1A-9	L0T1-2 L0T1-3	B Application	Buttons: Learning to categorise objects: Understand decomposition, pattern matching
1A-10	L0T1-2 L0T1-3	B Application	Thirty seconds: Learning activity to understand abstraction.
1A-11	L0T1-2 L0T1-3	B Application	Placemat: Practice to decompose spelling task and identify patterns
1A-12	L0T1-2	B Comprehension	MCQ
1A-13	L0T1-2	B Comprehension	MCQ
1A-14	L0T1-2	Application	MCQ
1A-15	L0T1-2	B Knowledge	MCQ
1A-16	L0T1-2	B Analysis	MCQ
1A-17	L0T1-2	B Comprehension and Knowledge	MCQ

Unit 2:

Number	Learning Outcomes
L0T2-1	Explain the importance of Algorithms in the Computational Thinking framework
L0T2-2	Write unambiguous, consistent algorithms
L0T2-3	Apply Logical thinking to a problem
L0T2-4	Appreciate how logic is used in algorithms to ensure validity
L0T2-5	Appreciate and discuss how computers do not think for themselves, they are an automation of our reasoning

Number	Name	Description	Assessment Type
2A-1	Week 2 MCQ1	Algorithms	GDrive Form Topic2_PreTest. MCQ
2A-2	Week 2 MCQ2	Algorithms	MCQ
2A-3	Week 2 MCQ3	Algorithms	MCQ
2A-4	Week 2 MCQ4	Logical thinking	MCQ
2A-5	Week 2 MCQ5	Logical thinking	MCQ
2A-6	Algorithm Puzzle	Playing Cards	Learning activity: Teacher Observation: Students practice card-tricks and observe characteristics of algorithms
2A-7	Algorithms_emoji	Algorithms: Write emoji instructions	Portfolio: Placemat Teacher notes the student evaluation of algorithm : Good, Bad etc, as they draw the students' emoji on the board.
2A-8	Logic Puzzle	Write algorithm:	Portfolio: Placemat
2A-9	Code Breaker	Code Breaker	Teacher Observation: Practice Logical Thinking. During reflection, students tell teacher how they did.
2A-10	Week 2 MCQ1	Algorithms	GDrive Form Topic2_POST_Test. MCQ
2A-11	Week 2 MCQ2	Algorithms	MCQ
2A-12	Week 2 MCQ3	Algorithms	MCQ
2A-13	Week 2 MCQ4	Logical thinking	MCQ
2A-14	Week 2 MCQ5	Logical thinking	MCQ

Activity	Learning Outcome	Bloom/Affective/Skill	Description
2A-1	L0T2-1	Knowledge	MCQ
2A-2	L0T2-1	Comprehension	MCQ

2A-3	L0T2-1	Knowledge	MCQ
2A-4	L0T2-4	Knowledge	MCQ
2A-5	L0T2-4 L0T2-5	Comprehension	MCQ
2A-6	L0T2-1 L0T2-5	PRACTICE: Following and algorithm	Playing cards
2A-7	L0T2-2	Application and Evaluation	Write an algorithm Placemat and teacher notes if students' algorithm achieved the desired results. Ask students if happy with result
2A-8	L0T2-3 L0T2-4	Application and Evaluation	Logic Puzzle: Placemat
2A-9	L0T2-3	Application, evaluate	Codemaster worksheets, Teacher report on student reflection on how they got on what they did right or wrong
2A-10	L0T2-1	Knowledge	MCQ
2A-11	L0T2-1	Comprehension	MCQ
2A-12	L0T2-1	Knowledge	MCQ
2A-13	L0T2-4	Knowledge	MCQ
2A-14	L0T2-4 L0T2-5	Comprehension	MCQ

Topic 3:

Number	Learning Outcomes
L0T3-1	Differentiate between Linear and Binary search
L0T3-2	Evaluate an algorithm
L0T3-3	Assess an algorithm
L0T3-4	Apply Computational Thinking skills to solve a problem

Number	Name	Description	Assessment Type
3A-1	Week 3 MCQ1	Linear & Binary Search	GDrive Form Topic3_PreTest. MCQ
3A-2	Week 3 MCQ2	Linear & Binary Search	MCQ
3A-3	Week 3 MCQ3	Linear & Binary Search	MCQ
3A-4	Divide and Conquer Puzzle	Puzzle Weighing scales	Learning activity: Students use divide and conquer to solve a puzzle
3A-5	Searching to speak	Two parts: Write an algorithm to establish a protocol for communication. See pattern matching and algorithms	Portfolio: Placemat Group work
3A-6	Searching to speak	Evaluate and assess their algorithm Use binary search	Portfolio: Placemat, Group work
3A-7	Jewel Activity	Binary search activity	Portfolio: Binary search
3A-8	Week 3 MCQ1	Linear & Binary Search	GDrive Form Topic2_POST_Test. MCQ
3A-9	Week 3 MCQ2	Linear & Binary Search	MCQ
3A-10	Week 3 MCQ3	Linear & Binary Search	MCQ

Activity	Learning Outcome	Bloom/Affective/Skill	Description
3A-1	L0T3-1	Knowledge + Comprehension	MCQ
3A-2	L0T3-1	Knowledge + Comprehension	MCQ
3A-3	L0T3-1	Application	MCQ
3A-4	L0T3-4	Application	Weighing puzzle
3A-5	L0T3-2 L0T3-3 L0T3-4	Create and Evaluation	Searching to speak 1

3A-6	L0T3-2 L0T3-3 L0T3-4 L0T3-1	Create and Evaluation	Searching to speak 2
3A-7	L0T3-1	Analysis and Application	Jewel Activity
3A-8	L0T3-1	Knowledge + Comprehension	MCQ
3A-9	L0T3-1	Knowledge + Comprehension	MCQ
3A-10	L0T3-1	Knowledge + Comprehension	MCQ

Topic 4:

Number	
LOT4-1	Explain Artificial Intelligence and the Turing Test
LOT4-2	Explain Artificial Intelligent machines are still written by humans
LOT4-3	Explore ethical dilemmas that impact us as both users of technology and also developers of technology
LOT4-4	Discuss the importance of Ethical Thinking when programming

Number	Name	Description	Assessment Type
4A-1	Week 4 MCQ1	Description of artificial Intelligence	GDrive Form Topic4_PreTest. MCQ
4A-2	Week 4 MCQ2	Description of Ethics	MCQ
4A-3	Week 4 MCQ3	Ethical Thinking	MCQ
4A-4	Week 4 MCQ4	Turing Test	MCQ
4A-5	Gender Bias Puzzle	Puzzle Weighing scales	Learning activity:
4A-6	Use CT to develop ChatBot	Create a ChatBot	Portfolio: ChatBot work
4A-7	Game of Tic Tac Toe	Highlight that AI programs are under the control of an intelligent being	Teacher Observation: Affective behaviour, are Computers Intelligent?
4A-8	Trolley problem	Reason their decision on an ethical dilemma	Teacher Observation: Student make a reactive decision. Will their decision change when they are “purchasing” a car.
4A-9	Ethical discussions	Discussion on data and ethics, ethics in driverless cars : not reactive so must be planned before hand	Portfolio Critical Thinking on this topic
4A-10	Week 4 MCQ1	Description of artificial Intelligence	MCQ
4A-11	Week 4 MCQ2	Description of Ethics	MCQ
4A-12	Week 4 MCQ2	Ethical Thinking	MCQ
4A-13	Week 4 MCQ2	Turing Test	MCQ

Activity	Learning Outcome	Bloom/Affective/Skill	Description
4A-1	L0T4-1	Knowledge + Comprehension	MCQ
4A-2	L0T4-1	Knowledge + Comprehension	MCQ
4A-3	L0T4-1	Knowledge + Comprehension	MCQ
4A-4	L0T4-1	Knowledge + Comprehension	MCQ
4A-5	L0T4-3 L0T4-4	Create and Evaluation	Gender Puzzle
4A-6	General practice of CT, L0T4-2, L0T4-1	Create and Evaluation	ChatBot: Portfolio
4A-7	L0T4-2	Knowledge + Comprehension + Evaluation	Tic Tac Toe
4A-8	L0T4-4	Evaluation	Trolley Problem
4A-9	L0T4-4, L0T4-3 , L0T4-2	Evaluation	Ethical Discussion: Portfolio
4A-10	L0T4-1	Knowledge + Comprehension	MCQ
4A-11	L0T4-1	Knowledge + Comprehension	MCQ
4A-12	L0T4-1	Knowledge + Comprehension	MCQ
4A-13	L0T4-1	Knowledge + Comprehension	MCQ

Topic 5

Number	
L0T5-1	Show how CT relates to programming
L0T5-2	Introduce pseudocode
L0T5-3	Explain the fundamentals of programming
L0T5-4	Introduce debugging

Number	Name	Description	Assessment Type
5A-1	Week 5 MCQ1	PseudoCode	GDrive Form Topic3_PreTest. MCQ
5A-2	Week5 MCQ2	Understand and debugging	MCQ
5A-3	Week 5 MCQ3	Understand and debugging	MCQ
5A-4	Debugging	Debugging	Learning activity: Application and reflection:
5A-5	Bad Logic	Bad Logic	
5A-6	Did I pass	Pseudocode: conditional, data- variable	Portfolio: Placemat, Group work Brennan and Resnick (2012)
5A-7	What is the average	Pseudocode: loops	Portfolio: Placemat, Group work
5A-8	What is the max	Loops, counters, conditional	Portfolio: Binary search Brennan and Resnick (2012)
5A-9	Cipher (optional)	Loops, counters, conditional	Brennan and Resnick (2012)
5A-10	Week 3 MCQ1	PseudoCode	GDrive Form Topic2_POST_Test. MCQ
5A-11	Week 3 MCQ2	Understand and debugging	MCQ
5A-12	Week 3 MCQ3	Understand and debugging	MCQ

Number			
L0T5-1		Show how CT relates to programming	
L0T5-2		Introduce pseudocode	
L0T5-3		Explain the fundamentals of programming	
Activity	Learning Outcome	Bloom/Affective/Skill	Description
5A-1	L0T5-2	Knowledge + Comprehension	MCQ
5A-2	L0T5-3 L0T5-4	Analysis and Application	MCQ
5A-3	L0T5-3 L0T5-4	Analysis and Application	MCQ
5A-4	L0T5-4	Application and Evaluation	debugging
5A-5	L0T5-4	Application and Evaluation	Bad logic
5A-6	L0T5-1 L0T5-2 L0T5-3	Create and Evaluation	Did I pass: application and reflection Bresnick (sequences)
5A-7	L0T5-1 L0T5-2 L0T5-3	Create and Evaluation	What is the average grade
5A-8	L0T5-1 L0T5-2 L0T5-3	Create and Evaluation	Thinking in twos : max and min
5A-9	L0T5-1 L0T5-2 L0T5-3	Create and Evaluation	Cipher (optional)
5A-1	L0T5-2	Knowledge + Comprehension	MCQ
5A-2	L0T5-3, L0T5-1	Analysis and Application	MCQ
5A-3	L0T5-3, L0T5-1	Analysis and Application	MCQ

Appendix M: Project Assessment

Having explored the concepts of:

Abstraction
Pattern Matching
Decomposition
Algorithms

You must explain how these concepts are relevant to computational thinking using a minimum of one practical example/activity.

The example or activity can be one we used in class and can be in the format of a logic puzzle, a card trick, or anything you deem acceptable.

What is expected in your submission?

- A title page with all relevant class details and an appropriate image for the subject
- Table of contents
- First Section (1 page) - Define 'Computational Thinking' in your own words and include a suitable image showing the concepts.
- Second Section - Explanation of activities/example to explain concepts. (1-2 pages per activity)
 - Activity Title
 - List of resources needed (Every activity should have some resource)
 - Outline/aim of activity
 - Step by step process of activity from start to finish (using images/sketches where possible)
 - Explanation of how activity demonstrates the concepts of computational thinking.
- References section

Appendix N: Workshop Questionnaire and Data

Workshop Content	N. Valid	N. Missing	Median	Mode
I understood the objectives of the workshop	5	0	5	5
The length of the workshop was appropriate	5	0	5	5
The workshop was informative	5	0	5	5
The examples presented helped me to understand the content	5	0	5	5
The workshop provided me with new information	5	0	5	5
Course Materials	N. Valid	N. Missing	Median	Mode
The Course Materials met your needs	5	0	5	5
The Course Materials were easy to use	5	0	5	5
Instructional Presentation	N. Valid	N. Missing	Median	Mode
The pace of the course was appropriate to adequately cover the content	5	0	5	5
The time given by the instructor to complete activities was appropriate	5	0	5	5
The instructor was knowledgeable about the subject	5	0	5	5
The instructor was responsive to my needs/questions	5	0	5	5
The instructor's presentation of the content was clear and informative	5	0	5	5
Course Content For Teaching	N. Valid	N. Missing	Median	Mode
The course content is appropriate for first year students	5	0	4	4
I understood how the course was designed	5	0	5	5
I understood the content of the course	5	0	5	5
The course content lends itself to the classroom setting	5	0	5	5
The course content meets its goal of assuming that no previous Computer Science experience is necessary for teachers	5	0	5	5
The course content meets its goal of assuming that no previous Computer Science experience is necessary for students	5	0	5	5
Course	N. Valid	N. Missing	Median	Mode
The course met your needs	4	1	5	5